

喻桃阳◎编著



IT需要简单化，其实编程很简单

学以致用、资源整合，开启成功之门

引导式学习，融合作者十年编程及培训经验

不再靠年轻吃饭，使每个程序员知道自己要做什么



浩为开发包(环境搭建所需工具)

各章范例源码+自我实践项目

清华大学出版社

PHP 快速入门与商用项目培训

喻桃阳　编著

清华大学出版社

北　京

内 容 简 介

本书针对 PHP 初学者及入门者，系统阐述 Web 开发的基本知识，结合数据库应用，使读者快速跨入 PHP 领域，对编程语言不再畏惧。HwCMS 以简洁的方式显示内容管理系统的基本功能，让您了解 PHP 开发的快捷；修改后的 UC Home 为浩为资源堂用户提供了便利，并基于软件开发的原则——尽量让用户使用方便。另外，还全面阐述了修改原因及过程。

PHP 与 Java 的整合，让您体味学以致用的最高境界，充分利用 PHP 的简单易学、开发高效和 Java 的强大功能及企业支持特性。PHP 负责 Web 层，Java 负责业务和数据逻辑层，形成 Web 开发的“黄金搭档”。

图书在版编目(CIP)数据

PHP 快速入门与商用项目培训/喻桃阳 编著 —北京. 清华大学出版社，2011.5
ISBN 978-7-302-25442-3

Ⅰ. P…　Ⅱ. 喻…　Ⅲ. PHP 语言—程序设计　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2011)第 078769 号

责任编辑：王　军　李维杰
装帧设计：孔祥丰
责任校对：胡花蕾
责任印制：
出版发行：清华大学出版社　　**地　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　编：**100084
社　总　机：010-62770175　　**邮　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　销：全国新华书店
开　本：185×260　　**印　张：**18.5　　**字　数：**416 千字
附光盘 1 张
版　次：2011 年 6 月第 1 版　　**印　次：**2011 年 6 月第 1 次印刷
印　数：1~4000
定　价：38.00 元

产品编号：

前　言

IT 专业技术是进入 IT 行业的“敲门砖”，要想成为 IT 人士，究竟需要掌握哪些技能呢？这方面的书籍数不胜数，不少书中的内容更让读者迷惑。读者感叹：IT 怎么这么难！

网络上流传着一张“软件程序员专业技能自检表”(见表 0-1)，这里有少许删减。

表 0-1　软件程序员专业技能自检表

基础项		熟练掌握？	有项目经验？	列入学习计划？
操作系统	Windows	□	□	□
	Linux	□	□	□
	UNIX	□	□	□
数据库	SQL Server	□	□	□
	Oracle	□	□	□
	DB2	□	□	□
基础知识	HTML 语言	□	□	□
	XML 语言	□	□	□
	数据结构常用算法	□	□	□
	UML 知识	□	□	□
开发语言和工具	VC++	□	□	□
	C++	□	□	□
	VS.NET	□	□	□
	Java	□	□	□
	其他请注明	□	□	□
软件过程	软件工程理论	□	□	□
	质量体系	□	□	□
	过程控制	□	□	□
	软件设计	□	□	□
	单元测试工具	□	□	□

(续表)

基 础 项		熟练掌握?	有项目经验?	列入学习计划?
软件过程	软件加密	□	□	□
	源代码控制	□	□	□
	说明书编写	□	□	□
	程序部署	□	□	□
项目组建设	编码规范	□	□	□
	源代码控制规范	□	□	□
	数据库设计规范	□	□	□
	公共组件或基类	□	□	□

若以此为参考标准，如果所列的技能大部分你都已经熟练掌握，那么恭喜你，你是一名合格的程序员！如果这些技能大部分你都已经拥有了项目经验，那么更要恭喜你，你是一名优秀的程序员！

但 IT 真有这么难吗？有人说一个稍微有点编程背景的普通人，只需要学习 PHP 半天时间，就可以上手开始开发 Web 应用；甚至一个从没有编程经历只做 Photoshop 的人，学两天 PHP，就能到处接活给人家开发网站，一个人全部搞定。以前有不少学习 JSP 的人问我，按照书上的介绍，他们无论怎么配置，都没法搭建开发环境。我一问过程，书上说要装 JDK、MySQL、Tomcat、Resin，甚至还要装 IIS、Apache。有必要这么复杂吗？对于一个初学者，不晕才怪。学了 HTML，开始学 Java；Java 的基础还没学完，发现 HTML 快忘了，于是赶紧学 HTML；学完 HTML，再学 JSP；等学完 JSP，又忘了不少 Java 基础知识，然后再回头看 Java。就这样不断循环，时间耗完了，头也转晕了，还是无法入门。

还有一个误区，有不少初学者或程序员，都在趋之若鹜地学习 SSH，似乎这些框架就成了一个人是否精通 Java，能否写 J2EE 程序的唯一事实标准和找工作的必备基础。然而，为什么要学习这些框架？这些框架的本质是什么？这些框架到底给你带来了什么？谁去思考过？

使用框架只是为了满足人们懒惰的要求，以提高工作效率。但是框架的使用，增加了初学者的难度，尤其是 SSH 更是增加了学习时间。框架只是为了解决在 Web 开发中所遇到的问题而诞生的，所以，千万不要为了学习框架而学习框架，而是要为了解决问题而学习框架，学以致用才是一个程序员的学习之道。初学者最重要的是尽快上手，上手之后，再去深入相对就会比较简单。

编程的核心是解决问题，语言只是问题解决的一种实现方式，而代码也只是项目开发中很少的一部分；当你有了解决问题的思路，即使不懂某种语言，也很快就能用其实现功能。不论是何种语言——Java、C++、PHP，还是数据库代码，基础知识均为数据类

型、数据结构、控制结构。大部分代码就是这三种内容的组合，再根据软件工程的思想来实现功能。

本书正是基于上述思想，参照 HwCMS、HoCAS 等实际项目，通过通俗易懂的描述，先讲述过程，塑造读者的问题解决能力，树立系统工程的理念，再讲述具体实现，在具体项目实践中理解核心知识点，从而帮助读者快速入门。

对于此书的学习，建议先看目录，了解书中讲了哪些点；再略读，你会发现前面一时不了解的知识，随着篇章的展开，不少知识自然而然就会了；最后细读，参照范例自己动手做。

优秀是种习惯，初学者应养成良好的编程习惯，树立好编程思想，懂得学以致用，而不要陷入代码的误区。任何技术大都免不了勤学苦练，“十年磨一剑”，希望本书能帮助读者实现理想，走向成功。

本书的完成得益于多位大学教师及大学生的积极参与。特别感谢上海华东师范大学软件学院杜育根副教授为本书提供指导和部分素材，更感谢上海对外贸易学院王朝晖教授对该书部分内容的修改。同时感谢陈聚雄、许童童、张艳伟等同学在本书写作过程中给予的建议。

本书是我在毕业 10 年、创业 8 年过程中在软件开发、员工培训、企业管理等方面的积累和总结，更有杜育根副教授从软件工程的高度对该书提供指导，力争为读者提供一种快速进入软件开发领域的途径。尽管如此，由于软件开发技术性太强，覆盖面太大，应用周期太长，涉及领域太多，开发方式和技术太繁杂，加之时间紧、水平有限，一定有许多不周到、不准确或存在错误之处，恳请读者提出批评和建议，并争取再版时修正。

浩为教育创始人兼 CTO　喻桃阳

目　　录

第 1 章

搭建开发环境

搭建开发环境，是进行数据库开发的前提。通过本章的学习，能使读者快速搭建一个开发环境。

由于开发环境配置繁杂，初学者一时难以理解，浩为将本书使用的软件整理成压缩文件，简称浩为开发包。浩为开发包提供完整的 Java、PHP、MySQL 运行及学习环境，解压即可使用，不用安装其他软件，自带 JRE、Web 服务器，整合 Tomcat、SQLEexplorer、jQuery 等插件，并提供思维导图软件 FreeMind，使你的思维更有逻辑性。

开发包 howwe1.rar 和 howwe2.rar 为 Eclipse 3.6 M6 版，已去掉 datatools，包含 Eclipse、PHP 5、Apache 2、JRE、HoCAT、FreeMind、Tomcat 插件、SQLExplorer 插件、jQuery 插件等软件及范例，近 130MB。纯 Java 开发包 work.rar 为 Eclipse 3.3 版，已去掉 datatools，包含 Eclipse、JRE、HoCAT、FreeMind、Tomcat 插件、SQLExplorer 插件、jQuery 插件等软件及范例，近 70MB。

浩为开发包不用安装其他软件，仅需按说明执行一下脚本即可使用。HoCAT 基于 Tomcat，提供运行监控、自带 JRE、Windows 服务管理等功能，是一款绿色软件，整合 Quercus，可同时运行 Java 和 PHP。Eclipse 作为多种编程语言的开发工具，已被广泛应用于采用 Java、PHP、C 等语言的开发实践中。

1.1 PHP 特征：请求无法持久

PHP，即 Hypertext Preprocessor，是一种在计算机上运行的脚本语言，用于处理动态网页，另外还包含命令行接口(command line interface)及图形用户界面(GUI)程序，但常用于处理动态网页。

PHP 最初简称为 Personal Home Page，是 Rasmus Lerdorf 为了维护个人网页、显示个人履历及统计网页流量，用 C 语言开发的一些 CGI 工具程序集，用来取代原先使用的 Perl

程序。他将这些程序和一些窗体解释器集成起来，称为 PHP/FI，可以和数据库连接，产生简单的动态网页程序。Rasmus Lerdorf 在 1995 年 6 月 8 日将 PHP/FI 公开，希望可以通过社区来加速程序开发和查找错误，这个版本被命名为 PHP 2。2004 年 7 月 PHP 5 发布，该版本强化面向对象功能、引入 PDO(PHP Data Objects，一个访问数据库的延伸库)，并有许多性能上的增强。2008 年，PHP 5 成了 PHP 唯一维护中的稳定版本。

PHP 的应用范围相当广泛，尤其是在网页的开发上。一般来说，PHP 大多运行在网页服务器上，通过运行 PHP 代码来产生网页。**PHP 是一种专门针对 Web 开发设计的语言，**对 Web 常见的需求都有内置的实现，加上容易学习的语法和简单的运行模型，造就了一大批 PHP 开发者。

特别提醒：PHP 是一种“**每次请求就是一个完整的生命周期**”的语言，每请求一次，所有相关对象必须重新加载，这些对象在请求一结束就灰飞烟灭，即 PHP 无法调用不同进程内的变量，简单说就是 PHP 无法持久。作为一种纯解释型语言，PHP 脚本在每次被解释时都进行初始化，在解释完毕后终止运行。运行时每一次请求都会创建一个单独的进程或线程，用来解释相应的页面文件。页面创建的变量和其他对象，都只在当前的页面内部可见，无法跨页面访问。PHP 的这种模型有非常好的优点，彻底杜绝了内存和资源的泄露，PHP 程序员再怎么菜，也极难写出有内存泄露的程序：请求一结束，所有的泄露都将被回收。所以 PHP 本身是追求简单、反框架的。

PHP 默认 opcode 不随进程持久，而是拥有和请求一样的生命周期。即请求一过来，PHP 解释器就编译所有的代码，生成 opcode，然后 Zend 虚拟机执行 opcode。请求一结束，opcode 也自动销毁。因此，PHP 可以实现一改代码无需重启进程就能看到效果。opcode cache(比如 APC、eAccelerator、XCache)所做的，只是把 opcode 缓存到进程的内存空间，进而避免了每个请求都编译一次代码所带来的开销。

小知识：PHP 脱颖而出的原因

PHP 是一种很优秀的开发语言，既可简单，也可复杂。不同的项目，可以用不同的风格。

1. 小项目：简单而直接的 PHP

功能页面在 20 个以下的网站，一般可以用很简单的框架结构来写。在这个规模级别，建议使用比较直接的面向过程的编码方法。原因很简单，因为没有必要把 class 文件弄得很多，大多在 controller 里边一个 new 就可以了。当然，需求频繁变化的项目必须分开。

开发小项目时 PHP 的优点很明显：快速开发，一目了然。缺点同时也被隐藏得很好。

2. 中型项目：结构优美的 OO 化的 PHP

中型项目的功能页面一般在 30 到 60 个，可以使用设计良好的框架来做。这个框架可以基于 MVC 模型，但必须封装了众多底层操作，一定要有一个基本且透明的 cache 机制。这样的话，为了适应变化而加入的 OO 机制就可以运行得更快更好。

这时 PHP 的缺点开始凸现，如对 OO 支持的不完整(虽然 PHP 5 有很大改进)、只能单线程模式、调试工具不够。优点：可快速开发，拥有广泛且可用的开源资源。

3. 大型项目：扩展、优化后的 PHP

大型项目一般指分布式项目，即程序需要被部署在 N 台服务器上。在这个规模级别，PHP 比 J2EE 确实缺乏很多支持。存在的问题不仅仅是 PHP 这个语言的问题，还包括了周边开发的问题：

- PHP 的页面代码共享，PHP 的源代码被载入内存后，就在其中保留。这个用 APC 和 Zend 的优化器可以搞定。
- PHP 页面之间如何共享数据对象？例如在 a.php 和 b.php 之间共享一个数据对象(如数组)，虽然可以用序列化来实现，但会有文件 IO；也可以用共享内存或 memcached 来处理。
- PHP 数据库连接池。在多访问的情况下，PHP 因为无法缓存数据库的连接而使数据库负荷很重，所以需要在数据库前加一个连接池，类似于 sqlrelay 的东西。另外，数据缓存也很重要，高访问压力下开发的提示：能不动数据库就不要动数据库。
- PHP 前端 cache 系统。一个透明可控的 cache 机制，能确保网站的页面以最少次数查询数据库，这个有很多实现。

一般的 PHP 应用，成功解决上述 4 大问题以后，就能应付稍微大些的访问量。这个级别上的应用，更多的是把 PHP 和 Java、C++、Python 等语言融合起来，使其成为一个高效系统。例如：我们可以用 memcached 做分布式内存管理，用 Lucene 做全文检索，用 EJB 容器放业务逻辑，而 PHP 却作为前端和系统的胶水，快速而灵活地把这些黏合起来。

总的来说，PHP 是一种易于学习和使用的服务器端脚本语言，只需要很少的编程知识，你就能使用 PHP 建立一个交互式的 Web 站点。只要了解一些基本的语法和语言特色，你就可以开始编程。甚至有人说：只要 30 分钟，你就可以将 PHP 的核心语言特点全部掌握。

PHP 借助 C++的语言形式(从语法上看，PHP 借鉴 C 语言的语法特征，是由 C 语言改进而来的)，引用类的概念，使得代码的可重复性应用变得异常简单。同时，PHP 开发者们为了使 PHP 更适合 Web 编程，开发了许多外围的基库，这些库就包含了很多更易用的层，从而可以方便实现各种功能。引用某牛人的一句话：PHP 可以做到你想让它做到的一切，而且无所不能!

1.2　PHP 运行环境

PHP 一般采用 http://www.php.net 提供的 PHP 5.3，由 Apache 2 整合 PHP 提供服务，常人很难一下就配好。在浩为开发包里已集成，详情请参考“**1.5　浩为开发包使用说明**”。

Resin 公司提供的 Quercus 框架，将 PHP 代码编译成 class，支持在 Java 虚拟机上直接

运行 PHP 应用，目前各大 Java 类 Web 服务器都已经支持这个框架。HoCAT 基于 Tomcat，提供运行监控、自带 JRE、Windows 服务管理功能，是一款绿色软件，整合了 Quercus，可同时运行 Java 和 PHP 程序。

Facebook 在 2010 年推出 HipHop 编译器，HipHop 把 PHP 源代码编译成 C++，以提高速度；根据 Facebook 内测，HipHop 的性能比对应的 PHP 版本高，CPU 负载能减少 30%。HipHop 属于自由软件，详情请参考 http://wiki.github.com/facebook/hiphop-php。

PHP 的语法参考了 Perl 和 C 语言，可以集成在 HTML 之中，下面示范一个简单的 Hello World 程序：

```
<?php
   echo 'Hello World!';
?>
```

PHP 解析引擎只解析“<?php”到“?>”之间的代码，不包含在“<?php”到“?>”之间的内容则会直接输出，因而可以用以下方式将 PHP 代码嵌入到 HTML 中：

```
<?php
//-PHP 程序代码
?>
html 内容
<?php
//-PHP 程序代码
?>
```

但是判断语句中的 HTML 代码则不会被直接输出：

```
<?php
if (false) {
?>
HTML Code
<?php
}
?>
```

PHP 可以使用 3 种注释形式：C、C++所使用的“/*...*/”和“//”，以及 Perl 的“#”。

1.3 Java 运行环境 JRE

Java 程序的运行要求必须安装 Java 运行环境，可以通过安装 JRE 或 JDK 获得。所有的 Java 代码首先被编译成后缀为.class 的类文件，然后类文件在虚拟机 JVM 上执行。

JRE(Java Runtime Environment，Java 运行环境)是运行 Java 程序所需环境的集合，包

含 JVM(Java Virtual Machine)标准实现及 Java 核心类库。

JDK(Java Development Kit)是 Java 开发工具包，里面包含了各种类库和工具，其中包含 JRE。

JVM 是一个虚拟的计算机，能模拟各种计算机的功能，它有处理器、堆栈、寄存器等虚拟的硬件架构，以及相应的指令系统。JVM 屏蔽了与具体操作系统平台相关的信息，使得 Java 语言编译程序只需生成可在 JVM 上运行的目标代码(字节码，也称类文件)，就可以在多种平台上平稳运行，而不需要修改，从而实现 Java 程序的跨平台特性。

JRE/JDK 可以从 http://java.sun.com/javase/downloads/index.jsp 下载，如果只是应用而不是开发，只下载 JRE 就足够。

1.4　Web 服务器

Tomcat 是一个轻量级应用服务器——Web 服务器，是开发和调试 JSP 程序的首选。

Tomcat 是由 Apache 软件基金会下属的 Jakarta 项目开发的一个 Servlet 容器，按照 Sun 提供的技术规范，实现了对 Servlet 和 JavaServer Page(JSP)的支持，提供的 Jasper 编译器能将 JSP 编译成相应的 Servlet。

Tomcat 本身内含一个 HTTP 服务器，一般可作为单独的 Web 服务器。Tomcat 通常与 Apache 或其他 Web 服务器一起工作。除了用于开发过程中的调试以及那些对速度和事务处理要求不高的用户外，将 Tomcat 单独作为 Web 服务器的用户在前几年并不多。但随着版本的更新，正在有越来越多的用户将其单独作为 Web 服务器，用于那些对速度和可靠性有较高要求的环境中。

Tomcat 使用 Java 开发，开放源代码，可以运行在任何一个装有 JVM 的操作系统之上。

下载地址：http://tomcat.apache.org/download-60.cgi。

小知识：什么是服务

Tomcat 是一个 Web 服务器，提供解析网页的服务。服务到底是什么呢？简单一点说，服务就是实现功能的程序，一般用于提供基于网络的应用。

服务的工作原理：监听某一端口，接收此端口的请求，返回相应的内容。如 Tomcat 监听端口 80(默认监听的端口是 8080)，接收 URL 请求后，返回相应的内容。

常用端口：

80：Web 服务端口，访问时，URL 地址隐含了默认的端口 80。如果不是 80，则必须在 URL 中添加端口号。很多协议都有默认的端口，如 FTP 为 21、SMTP 为 25。

139：由“NetBIOS Session Service”提供，主要用于提供 Windows 文件和打印机共享以及 UNIX 中的 Samba 服务。在 Windows 中，若要在局域网中进行文件的共享，就必须使用该服务。

1433：数据库 SQL Server 的默认端口。

3306：数据库 MySQL 的默认端口。

9002：数据库 HSQL 的默认端口。

在复杂的企业应用中，可能会在同一个服务器上安装多个厂家的产品，而这些产品中就可能使用多个数据库，如 MySQL。但为了管理方便，提供产品的厂家一般只使用自己的数据库，这样就必须在服务器上通过使用多个 MySQL 程序监听多个端口的方式来提供服务。

要测试某个端口号(如 139)是否可用，可在命令行(DOS 窗口)中输入“telnet 127.0.0.1 139”，如出现错误提示，则表示此端口号不通，如图 1-1 所示。



图 1-1　查看端口及服务

如果要查看所有端口，可在命令行(DOS 窗口)中输入“netstat –an”(参见图 1-1)。

其中：State 为 LISTENING，表示此端口正处于监听中，可提供服务；State 为 ESTABLISHED，表示两个 IP 地址之间已经建立连接。

木马程序的原理：先在计算机上用木马程序监听一个端口，即此端口的状态为 LISTENING，其他计算机通过网络访问此端口，建立一个连接(其状态为 ESTABLISHED)，即可任意操作被控制者的计算机。另一种方式是定时从某台计算机上获取指令，在计算机后台执行指令，例如定时访问某个页面(简称“刷票”)。

为了计算机的安全，建议将不用的端口全部关闭或者使用防火墙关闭对端口的访问。

1.5　浩为开发包使用说明

搭建开发环境是进行数据库开发的前提。开发环境配置繁杂，这给初学者带来了不少麻烦，浩为将本书使用的软件整理成压缩文件，简称浩为开发包(HwDep)。浩为开发包提供完整的 Java、PHP、MySQL 运行及学习环境，解压即可使用，不用安装其他软件，自带 JRE、Web 服务器，整合 Tomcat、SQLExplorer、jQuery 等插件，并提供思维导图软件

FreeMind，使你的思维更有逻辑性。

开发包 howwe1.rar 和 howwe2.rar 为 Eclipse 3.6 M6 版，已去掉 datatools，包含 Eclipse、PHP 5、Apache 2、JRE、HoCAT、FreeMind、Tomcat 插件、SQLExplorer 插件、jQuery 插件等软件及范例，近 130MB。下载地址为 http://zyt.howwe.net/me.php?574。

文件全部下载后，请放在 D:\根目录下，文件的解压如图 1-2 所示，图 1-3 为解压后目录 D:\howwe 下的部分内容，如有不同，请调整目录。也可直接复制随书光盘的 howwe 目录到 D:\根目录，再按提示进行安装。



图 1-2　howwe 解压

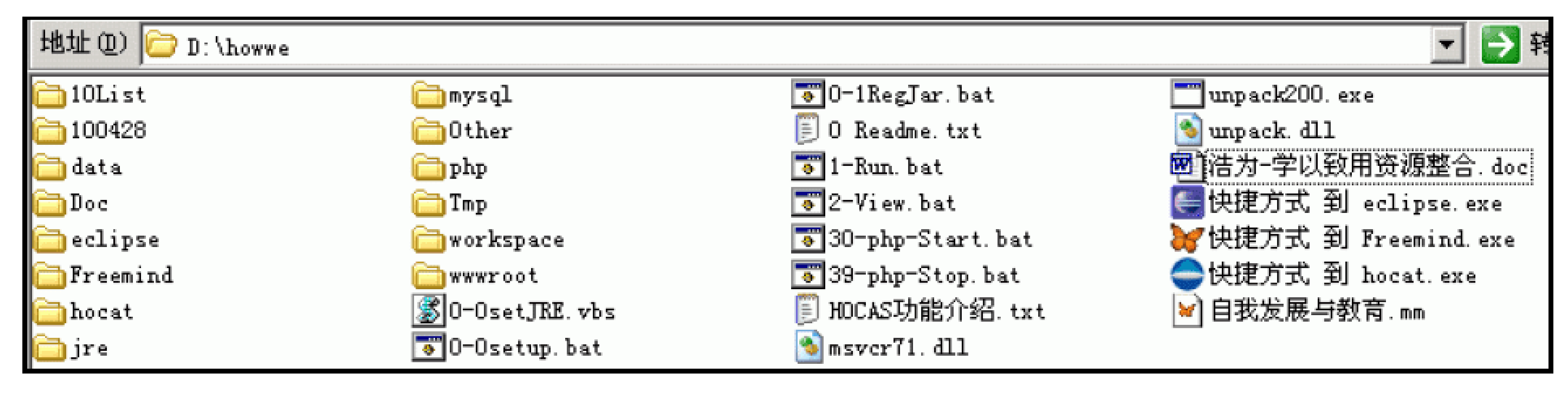


图 1-3　解压后的目录 D:\howwe

1. 安装步骤(D:\howwe 目录结构后面将详细介绍)

❖ **提示：**

如果已经安装 JDK，并且要使用已经安装好的 JDK，请删除目录 jre 并自己配置环境变量。对于初学者，请严格按照书中介绍的步骤，并且工作目录必须为 D:\howwe。

(1) 注册 Java 运行环境：有两种方式，自动配置或手工配置，任选一种即可。此步将配置两个环境变量：JAVA_HOME 和 path。

自动配置：双击 0-0setJRE.vbs 运行脚本，提示“set ok”即可。

手工配置：**资源管理器** => **我的电脑**，右击鼠标，选择**属性** => **高级** => **环境变量**。在图 1-4 所示的窗口中，先修改 path，在**变量值**中添加“;d:\howwe\jre\bin; ”；再**新建**变量，变量名为 JAVA_HOME，变量值为 D:\howwe\jre。

这两个值均可根据自己的配置进行修改，也可以修改文件 0-0setJRE.vbs 的相应路径后

运行。

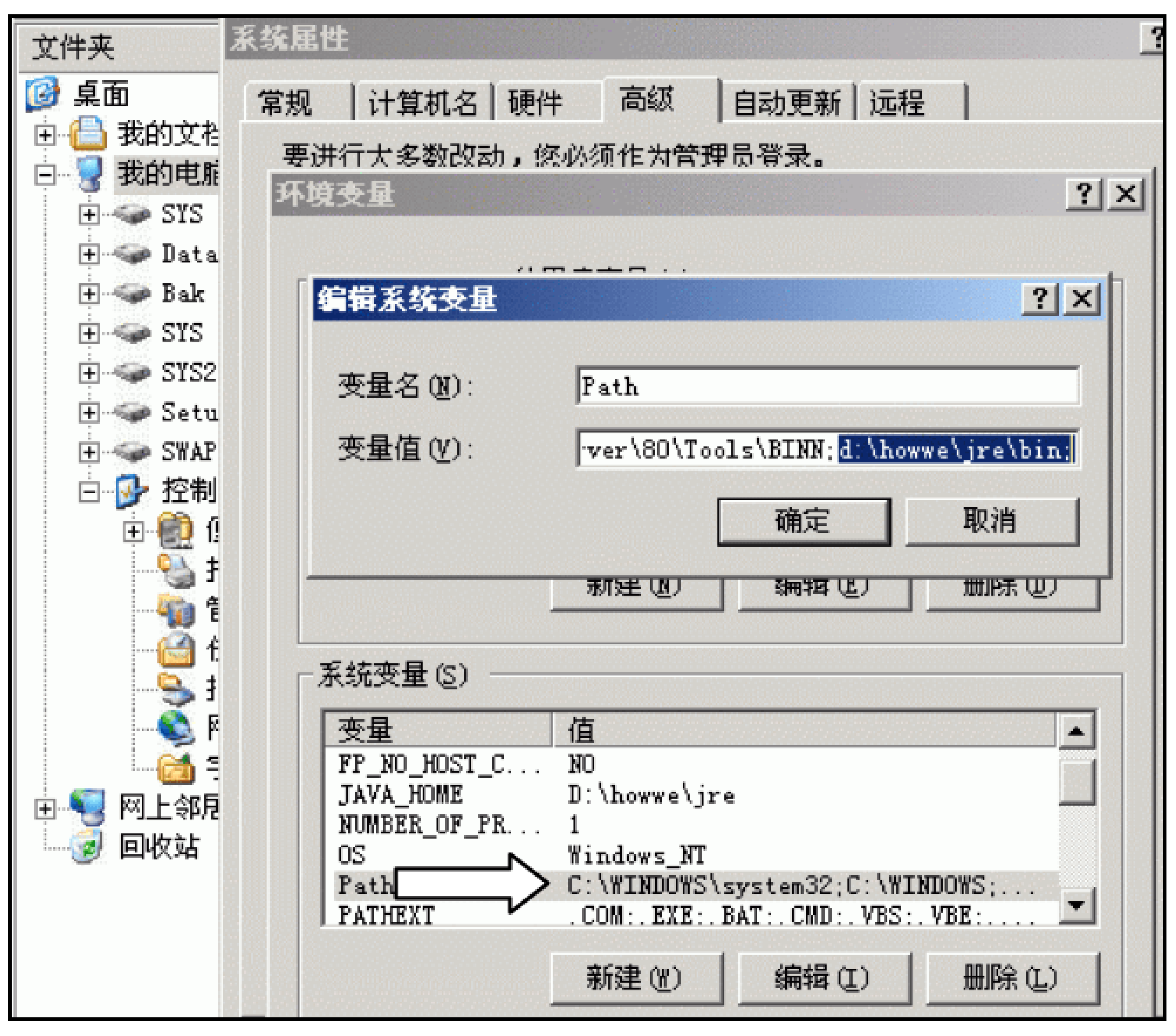


图 1-4　添加环境变量

(2) 注册服务及解压 Jar：运行(鼠标双击)D:\howwe\0-0setup.bat。先将 mysql(数据库)和 httpd(PHP 运行环境)注册为服务，再将 Jar 解压。

如图 1-5 所示，开始解压被压缩的运行程序，解压完成后，此窗口将自动关闭，0-0setJRE.vbs 和 0-0setup.bat 也将被删除。

```
C:\WINDOWS\system32\cmd.exe
"Howwe Develop Package" Reg Services, please wait ...
The service already exists!
The current server installed: d:\howwe\mysql\bin\mysqld.exe --defaults-file=d:\howwe\mysql\my.ini howwe-MySql
[SC] ChangeServiceConfig 成功
PHP...
[Tue May 04 07:38:24 2010] [error] howwe-WEB: Service is already installed.
[SC] ChangeServiceConfig 成功
```

图 1-5　文件解压

(3) Jar 运行关联：运行(鼠标双击)D:\howwe\0-1RegJar.bat，以关联 jar 及 jnlp 文件。即双击后能自动运行，此步仅限在需要双击 jar 文件就运行程序时执行。

2. 运行 Web 服务器(PHP 或 HoCAT 只需运行一个)

在单纯 PHP 运行环境下，运行 30-php-Start.bat，如图 1-6 所示，也可运行 php 目录下的 30-phpStart.bat；如要退出，可运行 39-phpStop.bat。

```
C:\WINDOWS\system32\cmd.exe
"Howwe Develop Package" Start..., please wait ...
howwe-MySql 服务正在启动 ...
howwe-MySql 服务已经启动成功。

howwe-WEB 服务正在启动 .
```

图 1-6　双击 30-phpStart.bat

在 Java&PHP 运行环境 HoCAT(修改自 Tomcat)下，运行 1-Run.bat 即可，也可参考 D:\howwe\hocat 下的“运行必读.txt”；如要退出，请直接单击**关闭**或 **close** 按钮。

如图 1-7 所示，用鼠标双击 D:\howwe\hocat\1 运行.bat 或 hocat.exe，以启动 Web 服务器 HoCAT(Tomcat 的包装版)。

也可以运行 D:\howwe 下的快捷键“**快捷方式 到 hocat.exe**”。

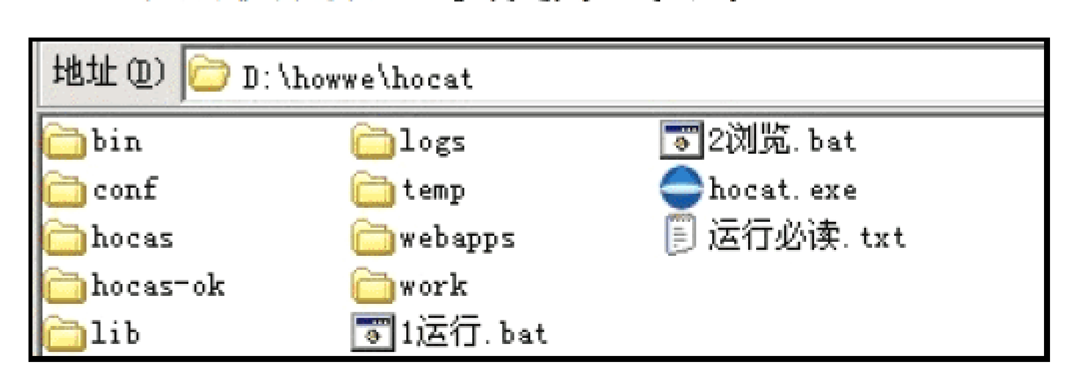


图 1-7　HoCAT 目录结构

hocas-ok 目录为《Java 快速入门与商用项目培训》(清华大学出版社，ISBN：978-7-302-24930-6)中的 Java 示范项目，hocas 为你将要完成的项目，hocas-ok 和 webapps\root 下有 PHP 和 Java 的整合范例。

3. 浏览网页

运行 D:\howwe\2-View.bat，将启用浏览器查看网页 http://127.0.0.1。如果运行的是 30-phpStart.bat，可显示图 1-8 所示的页面；否则，就是你的 80 端口已被占用。



图 1-8　纯 PHP 运行环境首页

如果运行的是 1-Run.bat，将显示图 1-9 所示的页面。否则，就是你的 80 端口已被占用。

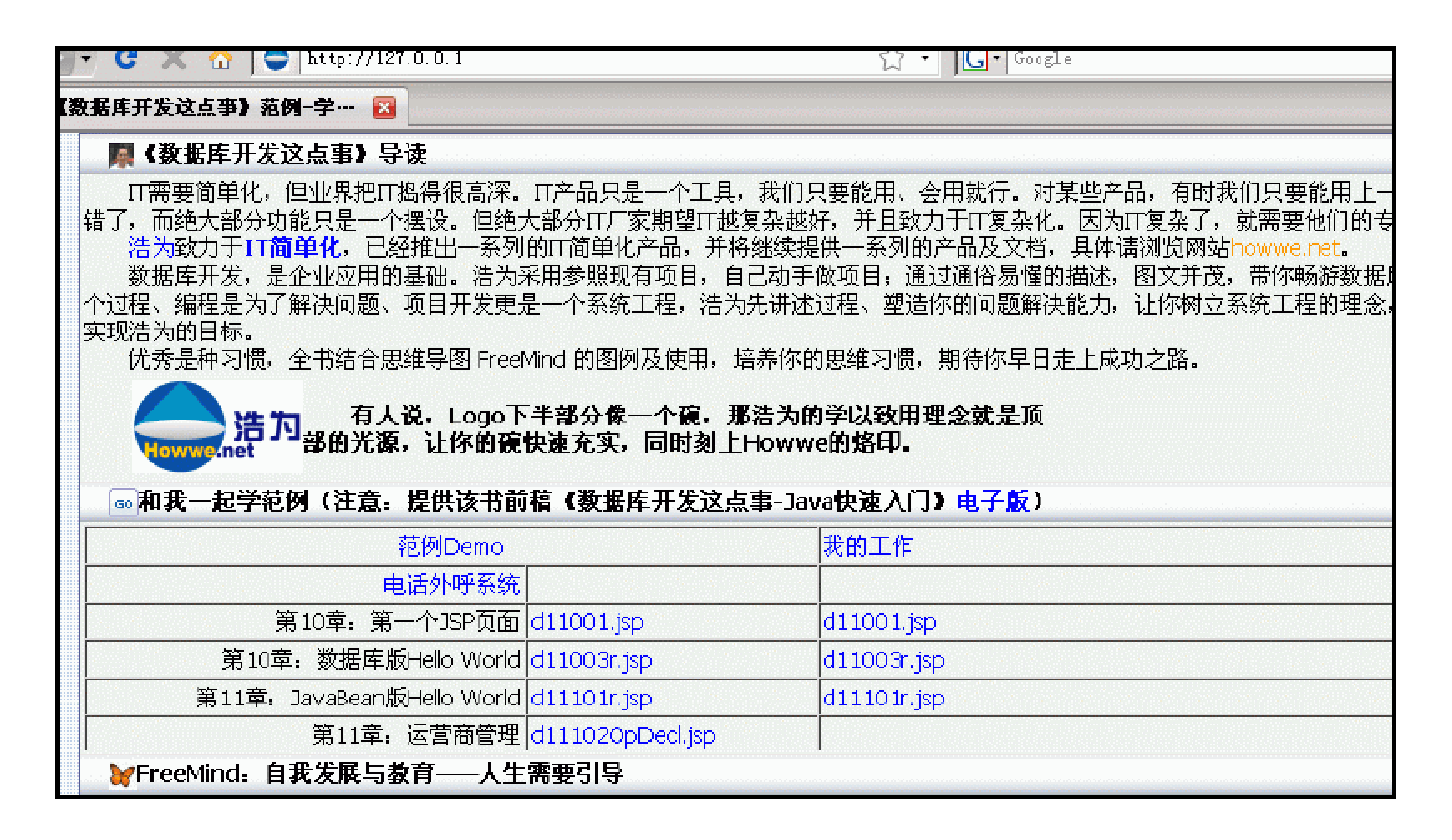


图 1-9　HoCAT 首页

也可以直接在 Eclipse 中运行 Web 服务器及浏览网页，具体请浏览 1.6 节。

❖ 注意：

1) 如果 PHP 环境的 80 端口已被占用，在文件 D:\howwe\php\Apache2\conf\httpd.conf 中，将所有的 80 改成你需要的端口，如 89，修改后请同时修改“2-View.bat”文件中的 80。

2) 如果 HoCAT 的 80 端口已被占用，在 D:\howwe\hocat\conf\server.xml 中查找 <Connector port="80"，将 80 改成你需要的端口，如 89，修改后请同时修改“2-View.bat”文件中的 80。

D:\howwe 为 HoCAT 上一级目录，可作为工作目录的范例。



图 1-10　howwe 目录结构

图 1-10 使用 FreeMind 生成，这是 FreeMind 的一项功能：将目录导入。

使用这一功能，可以更直观地显示目录及文件的结构，同时可以对目录或文件添加说明。如果要打开文件或目录，请移动鼠标至文件或目录前的图标上，待鼠标变为手形，即可打开。

❖ 注意：

尽量不要在 FreeMind 文件中删除文件或目录，因为有时可能会误删除，即在不知不觉中文件可能就没了。如想修改，最好将 mm 文件保存到另外的目录，即 mm 文件不能和导入的目录在同一目录或父目录中。

目录及文件说明：

- 10List：2010 年工作目录，存放备份的日工作内容，可用于事后整理及分析总结。
- 100428：2010 年 04 月 28 日的工作目录，记录当天的工作及相关的文档。需定时归档。
- Doc：存放常用的文档。
- data：存放数据库文件。
- Freemind：思维导图(Mind Map)软件，一种开源 Java 软件，可以帮助你整理思路，英国人托尼・巴赞创造的一种笔记方法，它以直观形象的图形建立起各个概念之间的联系。
- eclipse：Java、PHP、C 等编程语言的编辑工具，下节详细介绍。
- jre：Java 运行环境。
- hocat：Web 服务器，基于 Tomcat，自带 JRE，支持 Java 和 PHP。
- Other：存放其他文件。
- mysql：MySQL 数据库运行程序 v5.1.40 及管理软件 SQL-Front。
- Tmp：存放临时文件。
- php：PHP 运行环境。
- wwwroot：PHP 运行环境存放文件的根目录。
- workspace：Eclipse 的工作目录。

在日常工作中使用这种结构的目录，将有助于工作的顺利开展，形成一种良好的工作习惯。比如日常文件的管理，帮助养成按日期存放及整理的习惯。以后要找文件时，便不会找不着。

1.6 Eclipse 使用简介

Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台。用户一般将 Eclipse 当

作 Java IDE 来使用，用于编辑 Java 代码(JSP、JSF 也可归属于 Java 代码)，也可以编辑 PHP、C 等语言代码。

下载地址为 http://www.eclipse.org/downloads/。

如果想自己配置 Eclipse，步骤如下(建议直接使用 1.5 节介绍的开发包，以便快速上手)：

(1) 选择 **Eclipse IDE for Java EE Developers**，2010 年 4 月时的版本为 3.5.2，下载文件为 eclipse-jee-galileo-SR2-win32.zip。

(2) 下载完之后，将文件 eclipse-jee-ganymede-SR2-win32.zip 复制到目录 D:\howwe 下，右击该文件，如图 1-11 所示，选择**解压到当前文件夹**，解压后的文件全部在目录 D:\howwe\eclipse 下。

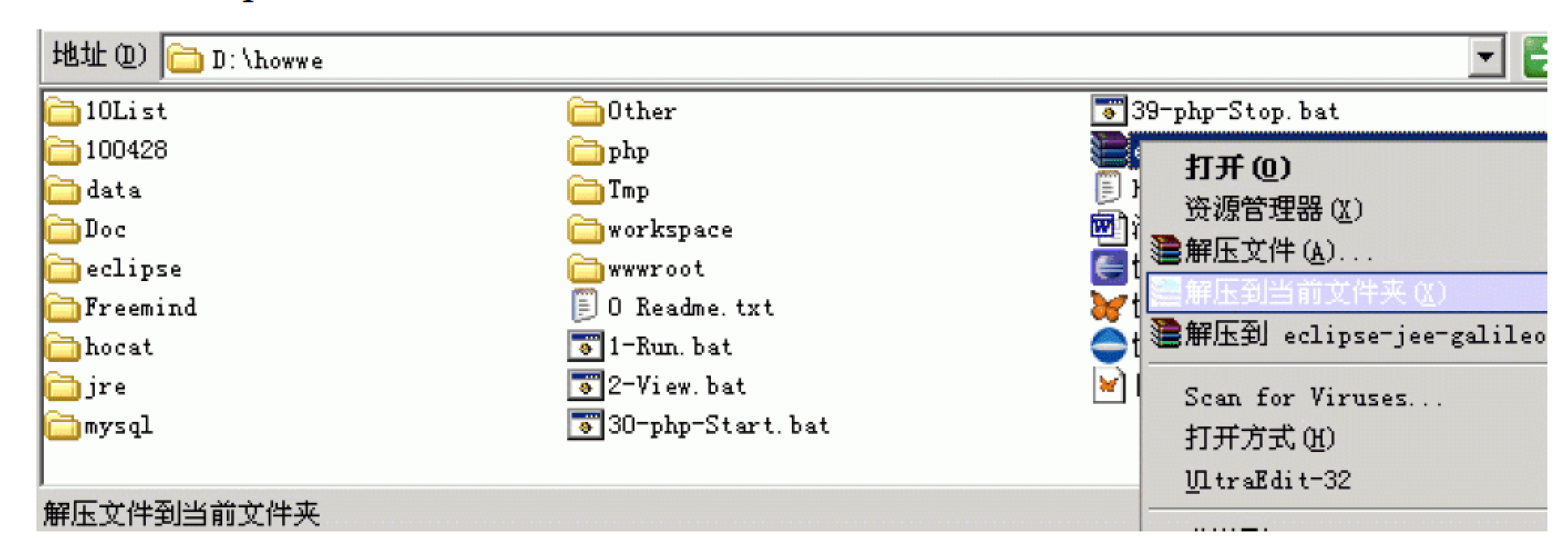


图 1-11 解压 Eclipse

❖ **注意：**

若已经存在 eclipse 目录，解压时请选择“全部覆盖”。目录下的文件将在后面用到。

双击 D:\howwe\eclipse 目录下的 eclipse.exe 文件，也可以双击 D:\howwe 下的“快捷方式到 eclipse.exe”。由于 Eclipse 将加载很多插件，如果运行的计算机配置不高，则可能需加载几分钟才能完成。第一次运行 Eclipse 时，将出现图 1-12 所示的选择 Workspace 的界面，请选择 D:\howwe\workspace，此目录下有 Eclipse 的配置文件及项目信息，便于初学者使用。

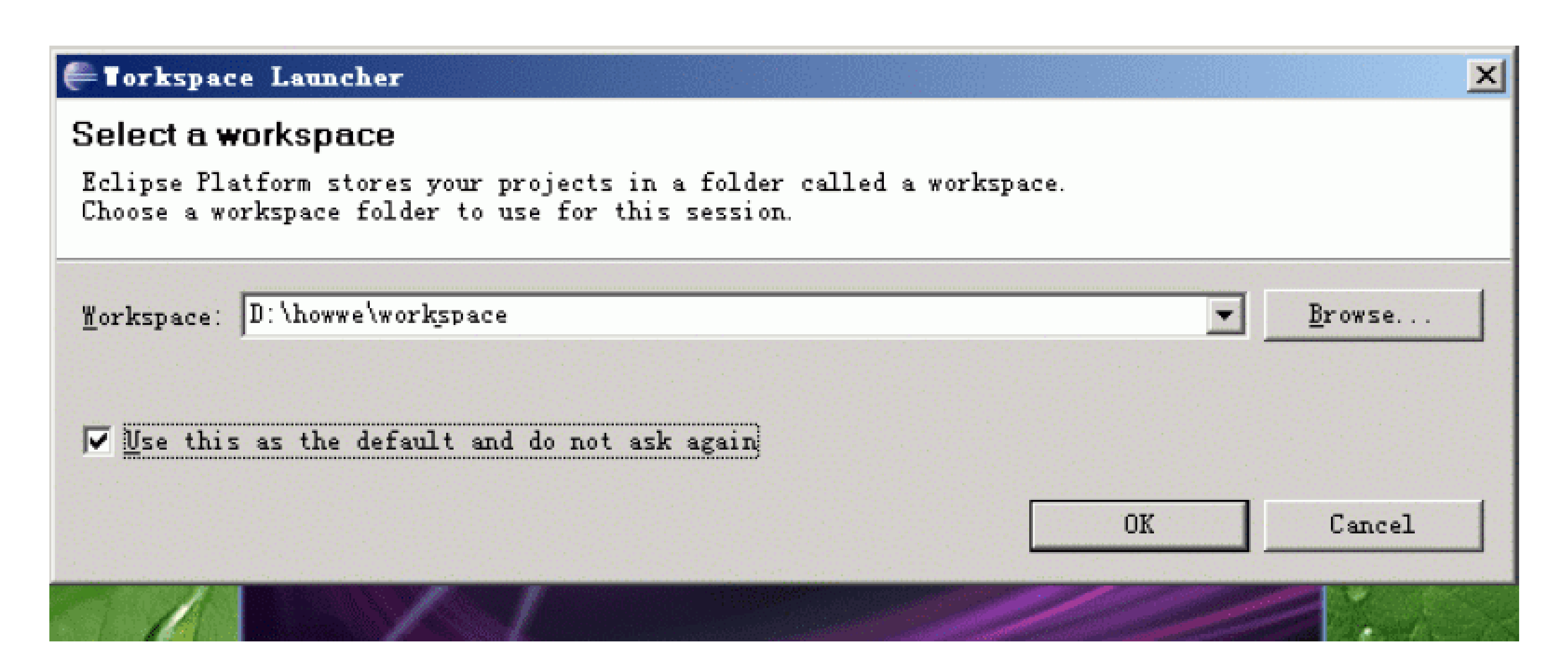


图 1-12 选择 Workspace

❖ 注意：

1) 使用整合包不会出现图 1-12 所示的界面。

2) 不推荐使用 MyEclipse，此软件太占用系统资源，会导致很多操作不流畅。

Eclipse 启动成功，界面如图 1-13 所示(注意：编辑器区域打开的文件与图可能不一致)。

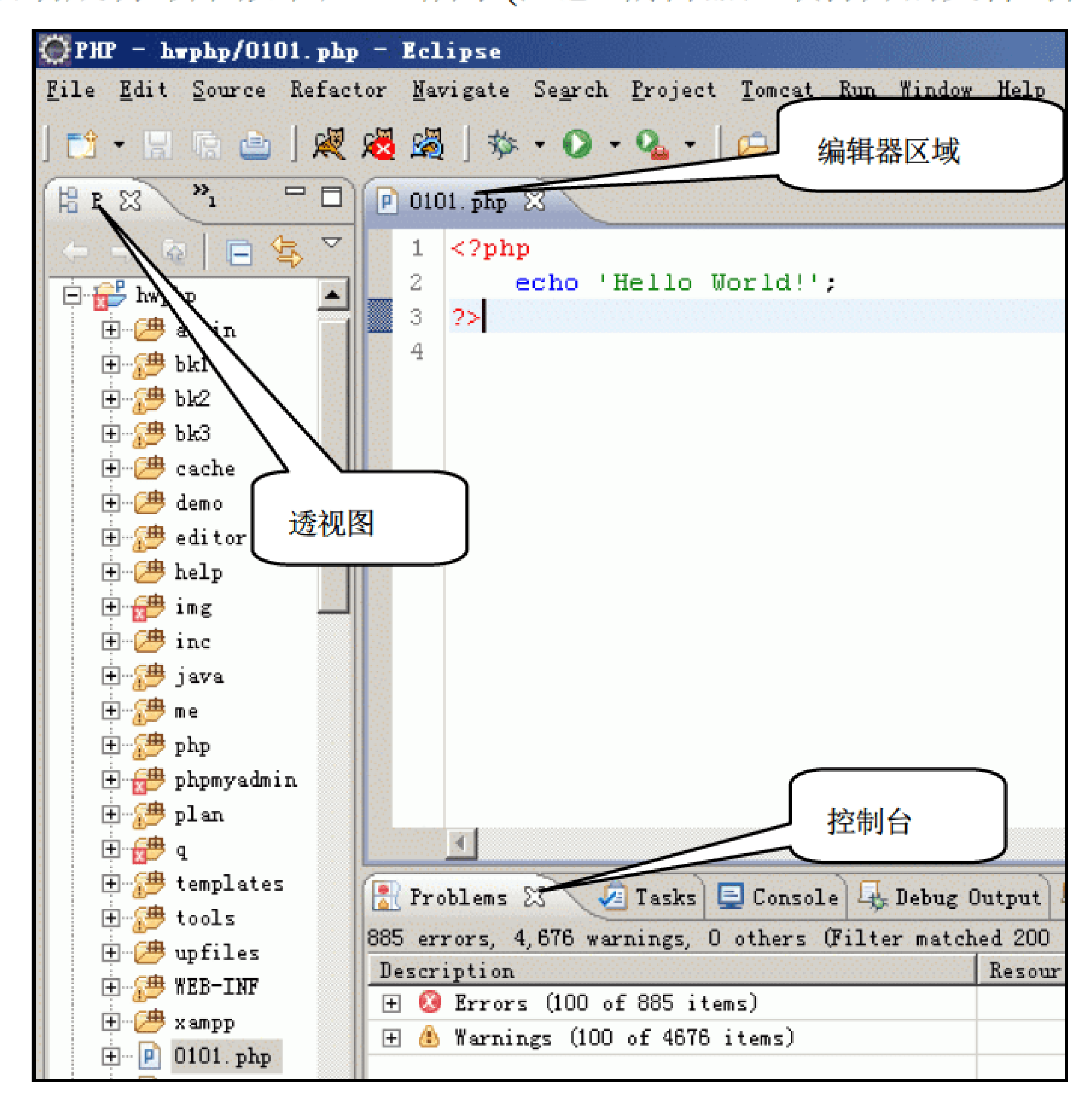


图 1-13　Eclipse 的工作界面

❖ 注意：

如果 Project 视图区的 PHP 示范项目 hwphp 显示为红叉，则表示此项目有错误。但如果是 HTML 文件上的提示，则不一定是错。按 F5 键刷新即可。

Eclipse 工作台由几个称为**视图(view)**的窗格组成，比如左上角的 Project 视图。窗格的集合称为**透视图(perspective)**。

Project 视图允许你创建、选择和删除项目。

右侧的窗格是**编辑器区域**，用于编辑文件，如图 1-13 所示的文件 0101.php。

右下角区域为**控制台**，控制台里面的部分称为标签，用于显示和项目相关的内容及加载的插件。常用的标签如下：

- Problems：项目编译后的提示信息，包括提示、错误等。
- Progress：工作提示窗口，如 Eclipse 对项目的编译、插件的更新等。
- Search：查询结果。

● Console：Tomcat 运行日志。

小知识：Eclipse 能做什么

Eclipse 平台是一个具有一组强大服务的框架，该框架支持插件(比如 JDT)和插件开发环境(PDE)。它由以下几个主要部分构成：平台运行库(Platform Runtime)、工作区(Workspace)、工作台(Workbench)、团队支持(Team)和帮助(Help)，如图 1-14 所示。

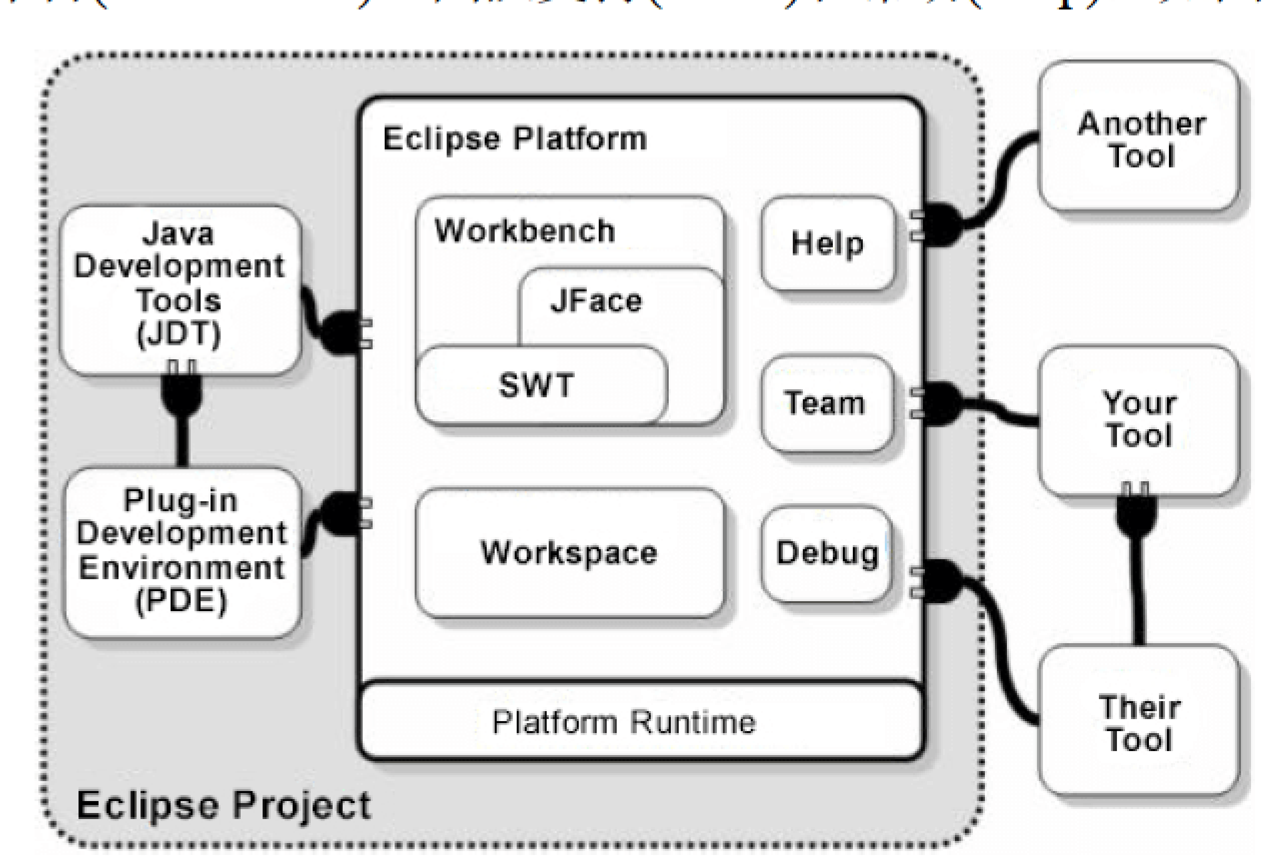


图 1-14　Eclipse 平台的构成

平台运行库

平台运行库是内核，它在启动时检查已安装了哪些插件，并创建关于它们的注册表信息。为降低启动时间和资源使用，它在实际需要时才加载该插件。除了内核外，其他每样东西都是作为插件来实现的。

工作区

工作区是负责管理用户资源的插件，包括用户创建的项目、项目中的文件，以及文件变更和其他资源。工作区还负责通知其他插件关于资源变更的信息，比如文件的创建、删除或更改。

工作台

工作台为 Eclipse 提供用户界面。它是使用标准窗口工具包(SWT)和一个更高级的 API(JFace)构建的；SWT 是 Java 的 Swing/AWT GUI API 的非标准替代者，JFace 则建立在 SWT 基础上，提供用户界面组件。

团队支持

团队支持组件负责提供版本控制和配置管理支持。它根据需要添加视图，以允许用户与所使用的任何版本控制系统(如果有的话)交互。大多数插件都不需要与团队支持组件交互，除非它们提供版本控制服务。

帮助

帮助组件具有与 Eclipse 平台本身相当的可扩展能力。与插件向 Eclipse 添加功能相同，帮助组件提供一个附加的导航结构，允许工具以 HTML 文件的形式添加文档。

Eclipse 是一个成熟的、精心设计的可扩展体系结构，就其本身而言，它只是一个框架和一组服务，可通过添加插件组件构建开发环境。Eclipse 附带了一个标准的插件开发环境(Plug-in Development Environment，PDE)，包括 Java 开发工具(Java Development Tools，JDT)。

体系结构的详细内容，请参考《Java 快速入门与商用项目培训》的“第 19 章 结构、MVC、框架”。

Eclipse 的设计思想是：一切皆插件。Eclipse 核心很小，其他的所有功能都以插件的形式附加于 Eclipse 核心之上。 Eclipse 的基本核心包括：图形 API (SWT/JFace)、Java 开发环境外挂程序(JDT)、外挂程序开发环境(PDE)等。

插件是功能单元的别称。Eclipse 平台运行时，先在插件子目录中名为 plugin.xml 的文件中查找插件的声明(称为**插件清单**)，这些子目录位于 Eclipse 目录下名为 plugins 的公共目录(具体而言，就是 eclipse\plugins)下。

Eclipse 的基础是富客户平台(Rich Client Platform，RCP)。RCP 包括下列组件：

- 核心平台(激活 Eclipse，运行外挂程序)
- OSGi(标准集束框架)
- SWT(可移植构件工具包)
- JFace(文件缓冲、文本处理、文本编辑器)
- Eclipse 工作台(即 Workbench ，包含视图(views)、编辑器(editors)、视角(perspectives)和向导(wizards))

Eclipse 的 RCP 框架还可用来作为与软件开发无关的其他应用程序类型的基础，比如内容管理系统。后续章节用到的 SQLExplorer ，其独立客户端(Standalone Client)版本就基于 RCP，在 Eclipse 里用的版本为 Eclipse Plugin。SQLExplorer 可以通过 JDBC 访问几乎任何一种数据库，同时还支持用 Hibernate 这样的工具访问数据库。

1.7 新建 Eclipse 项目

1. 项目导入

操作步骤：**File** => **Import**(中文版对应为**文件** => **导入**)，在图 1-15 所示的界面中选择 **General** => **Existing Projects into Workspace**(中文版对应为**标准** => **现有项目导入工作空间**)，单击 **Next**(中文版对应为**下一步**)。

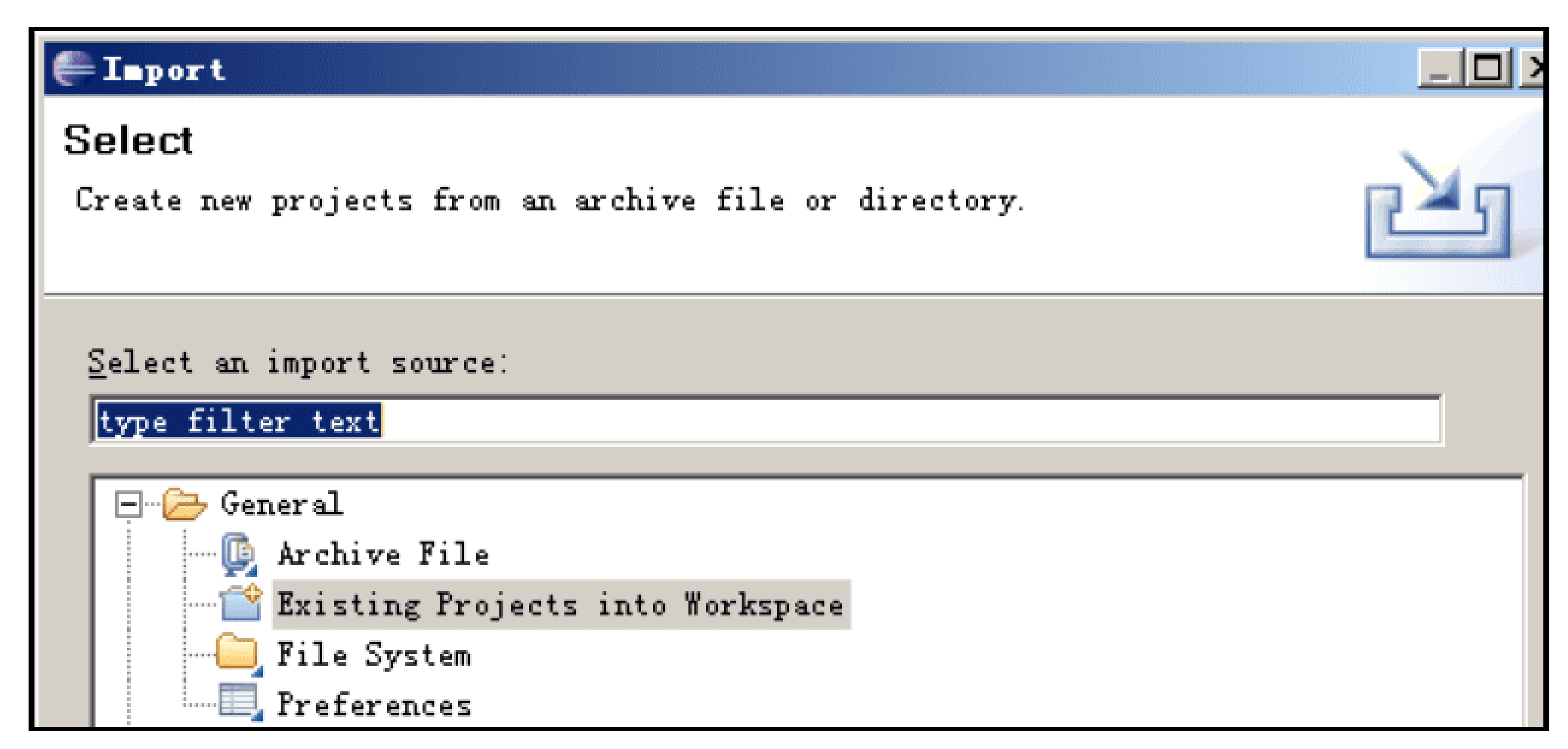


图 1-15 在 Eclipse 中导入项目

在图 1-16 中，单击按钮 **Browse**(中文版对应为**浏览**)，再选择项目目录 D:\howwe\hocat\hocas-ok，因为 howwe-ok 已经存在，故不能再导入。如果要导入的项目没有加载过，选中要加载的项目后单击按钮 **Finish** (中文版对应为**完成**)即可。

❖ 注意：

如果项目 howwe-ok 已经存在，就不能再导入。项目导入时，将先检查已存在的项目名，如果已经存在，就不会在图 1-16 所示的项目列表框(**Projects:**下方)中列出。

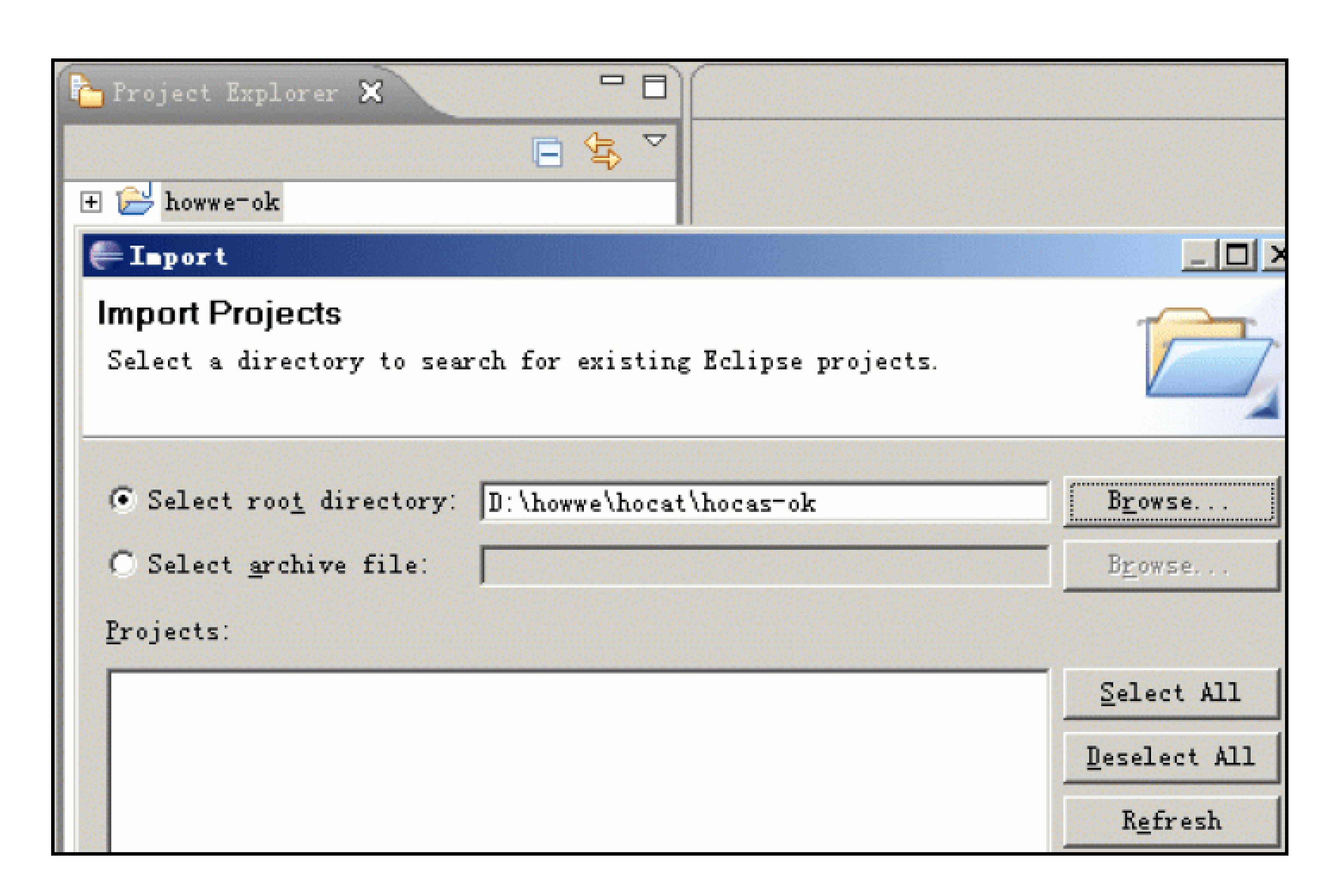


图 1-16 选择项目目录

2. 新建项目

操作步骤：

(1) **File** => **New** => **PHP Project**(中文版对应为 **文件 => 新建** => **PHP 项目**)，如果 **New** 下面没有 **PHP Project**，请选 **Project**，在出现的窗口中选择 **PHP** => **PHP Project**。

(2) 在图 1-17 所示的窗口中，在项目名 **Project name** 处输入 me，选择 **Create project from existing source**，单击 **Browse** 按钮，选择目录 D:\howwe\wwwroot\me(目录 me 需新

建)后单击 **Next**，最后单击 **Finish**。

❖ **注意：**

如果选择 **Create new project in workspace**，将在工作区 workspace 下自动创建一个以项目为名称的目录。

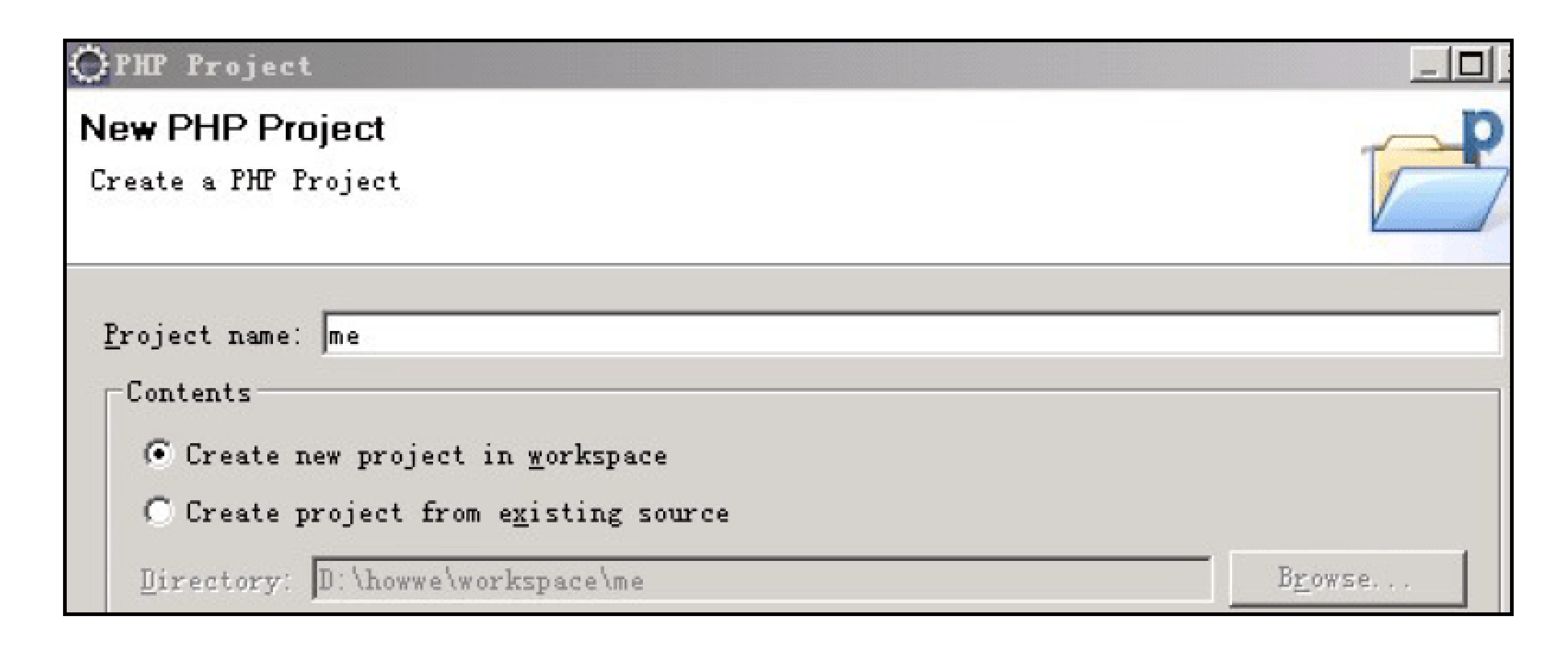


图 1-17　新建项目

新建的项目 me 如图 1-18 所示。

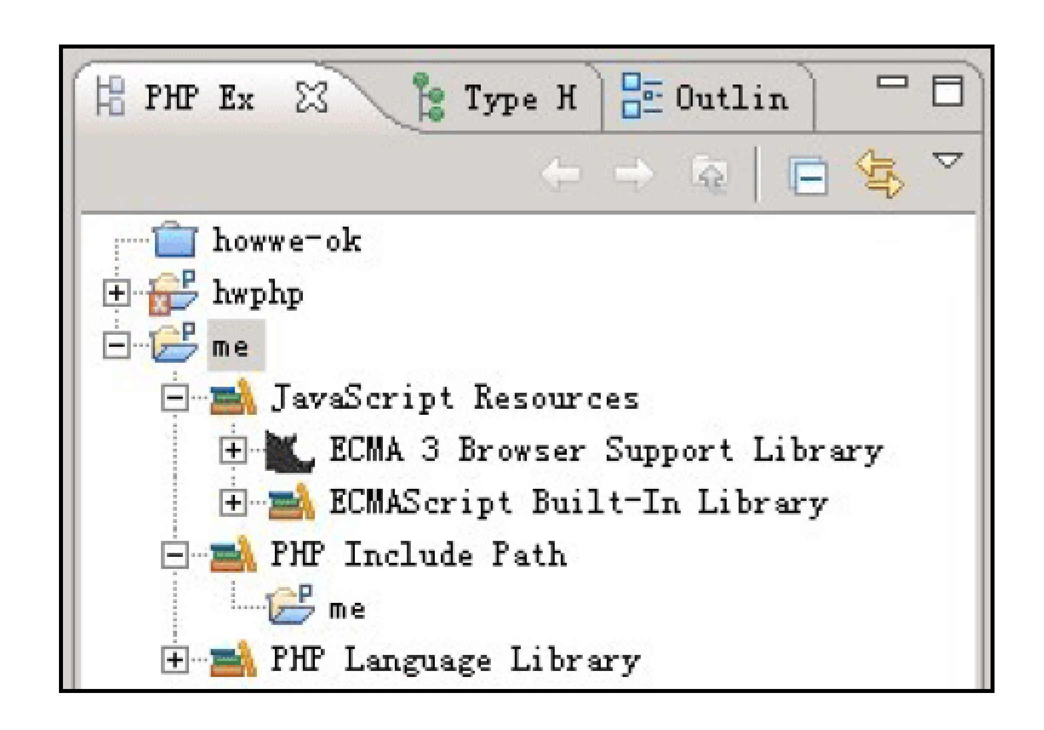


图 1-18　新建的项目 me

第 2 章

激发你的思维潜能

思维导图是用图形化的方式展现想法和概念，作为一个可视化的思考工具，它有助于信息的结构化，便于更好地分析、理解、综合、回忆及产生新的想法。正像所有的好点子一样，它的能力就在于它的简单，其中 FreeMind 是思维导图的典范。FreeMind 的使用特别简单，以至于我都不想用太多的篇章来介绍其用法。

形象思维是指运用头脑中积累起来的表象进行的思维。表象是我们以前知觉过的，而在头脑中再现的那些对象的映象。即现实中的苹果或描述的苹果是对象，通过形象思维处理过后，在我们的大脑中形成苹果的映象。

随着思维导图使用增多，思考问题的条理性大有提高，在无形中就提高了你的思考能力，更激发了你的思维潜能。

使用 FreeMind 做做日程管理、事前计划，并安排日程及工作，事后及时跟踪，将培养你自己持续努力的习惯，永葆干劲十足。

人有偷懒的天性，面对漫长的人生之路，很多人的失败就在于不能持之以恒。人必须认清自己，方能激发自己的潜力，使用 FreeMind 软件或思维导图的思想来分析问题及实现“日程管理”，坚持一段时间，就能发现自己的思维习惯及日程等大有改善。有人说，优秀是种习惯。人生需要引导，结合思维导图 FreeMind 等的使用，辅以日程事务管理，培养你的思维习惯，期待你早日走上成功之路。

❖ 提示：

形象思维、逻辑思维等概念可参考《Java 快速入门与商用项目培训》中的“**2.6 思维导图培养你的形象思维能力**”或自己去 Google。

2.1　我的思维导图使用历程

几年前我就知道思维导图，曾使用 FreeMind 做过一些分析，但一直没有细究，更没有深入使用。2008 年底，我请一朋友帮我对公司的定位做分析，我先列出 4 大方向，他接着分析各方向的优劣，讨论后确定公司以培训为主，所以 2009 年转向 IT 培训。

在写书之前，做了一些产品，其中就包含 FreeMind 修改版。因为我觉得这个软件不错，值得推荐给别人，但还没有上升到能提高思维能力的层次。

在做“HwCall 电话外呼系统-技术方案”时，使用 FreeMind 做了几个系统构架图，觉得这个软件很不错。于是在撰写本书时开始使用 FreeMind，尤其是在分析“自我发展与教育的关系”(人生需要引导相关概念的第一版)时，突然意识到那朋友给我做公司定位分析时，其中就包含思维导图的思想。

这样我就对思维导图的重要性有了更深的认识，于是在书中大量使用 FreeMind，同时也查了不少思维导图的资料，以至于在该书中用一章来介绍思维导图。

在 QQ 群 78928780 中，特意要求群员使用 FreeMind 为自己做一个“近期学习安排”，即后面提到的“日程管理”，做过的人都觉得对他们很有帮助。

使用思维导图几个月后，发觉自己思考问题的条理性大有提高。例如在思考某论坛的改版时，我想到的是哪些人会来论坛，而论坛能为这些人提供哪些内容，列出了几大人群，觉得论坛应该从这个角度去考虑改版，即论坛的改版先得考虑定位。

暑假期间，我同时在安排人参与《软件工程术语中英对照案例解析教程》的编写，经常和我的老师交流，我最关心的是教程的定位，即哪些人会来看这本书。这样很快就抓住了问题的本质，从而将问题解决得更加完美。

2.2　思 维 导 图

有人说思维导图让你在一天之内就能分析几本书并牢记重点！随着思维导图使用增多，思考问题的条理性大有提高，在无形中就提高了你的思考能力，更激发了你的思维潜能。

思维导图是一种革命性的思维工具。思维导图采用图形化的方式展现想法和概念。作为一个可视化的思考工具，它有助于信息结构化，以便于更好地分析、理解、综合、回忆及产生新的想法。正像所有的好点子一样，它的能力就在于它的简单，却又极其有效！

与传统的记笔记或直线型文字不同，思维导图里的信息完全可以像大脑实际思考过程那样进行组织。思维导图兼具分析性和艺术性，会让大脑思维更为发散，可以激发所有认知功能。同时，最棒的地方是思维导图非常有趣。

19 世纪 60 年代，英国著名心理学家托尼・巴赞在研究大脑的力量和潜能过程中，发

现艺术家达·芬奇在他的笔记中使用了许多图画、代号和连线。他意识到，那就是达·芬奇拥有超级头脑的秘密所在。在此基础上，巴赞发明了思维导图这一风靡世界的思维工具。

思维导图作用概括：增强使用者的记忆能力、增强使用者的立体思维能力(思维的层次性与联想性)、增强使用者的总体规划能力。

为什么思维导图功效如此强大？道理其实很简单。

首先，它基于对人脑的模拟，它的整个画面像一个人大脑的结构图(分布着许多“沟”与“回”)。

其次，这种模拟突出了思维内容的重心和层次。

再次，这种模拟强化了联想功能，正像大脑细胞之间无限丰富的连接。

最后，人脑对图像的加工记忆能力大约是文字的 1000 倍。

思维导图是一种创造性的、有效的记笔记的方法，能够用文字将你的想法“画出来”，让你更有效地把信息放进你的大脑，或是把信息从你的大脑中取出来。

所有的思维导图都有一些共同之处：它们都使用颜色，它们都有从中心发散出来的自然结构，它们都使用线条、符号、词汇和图像，遵循一套简单、基本、自然、易被大脑接受的规则。使用思维导图，可以把一长串枯燥的信息变成彩色的、容易记忆的、有高度组织性的图画，它与我们大脑处理事物的自然方式相吻合。

尽管在刚开始的时候看起来太乱，不过一旦你放弃自己根深蒂固的写线性笔记的习惯，你就不会回头了。再看看有人用思维导图做的人生引导图(参见图 2-1)。

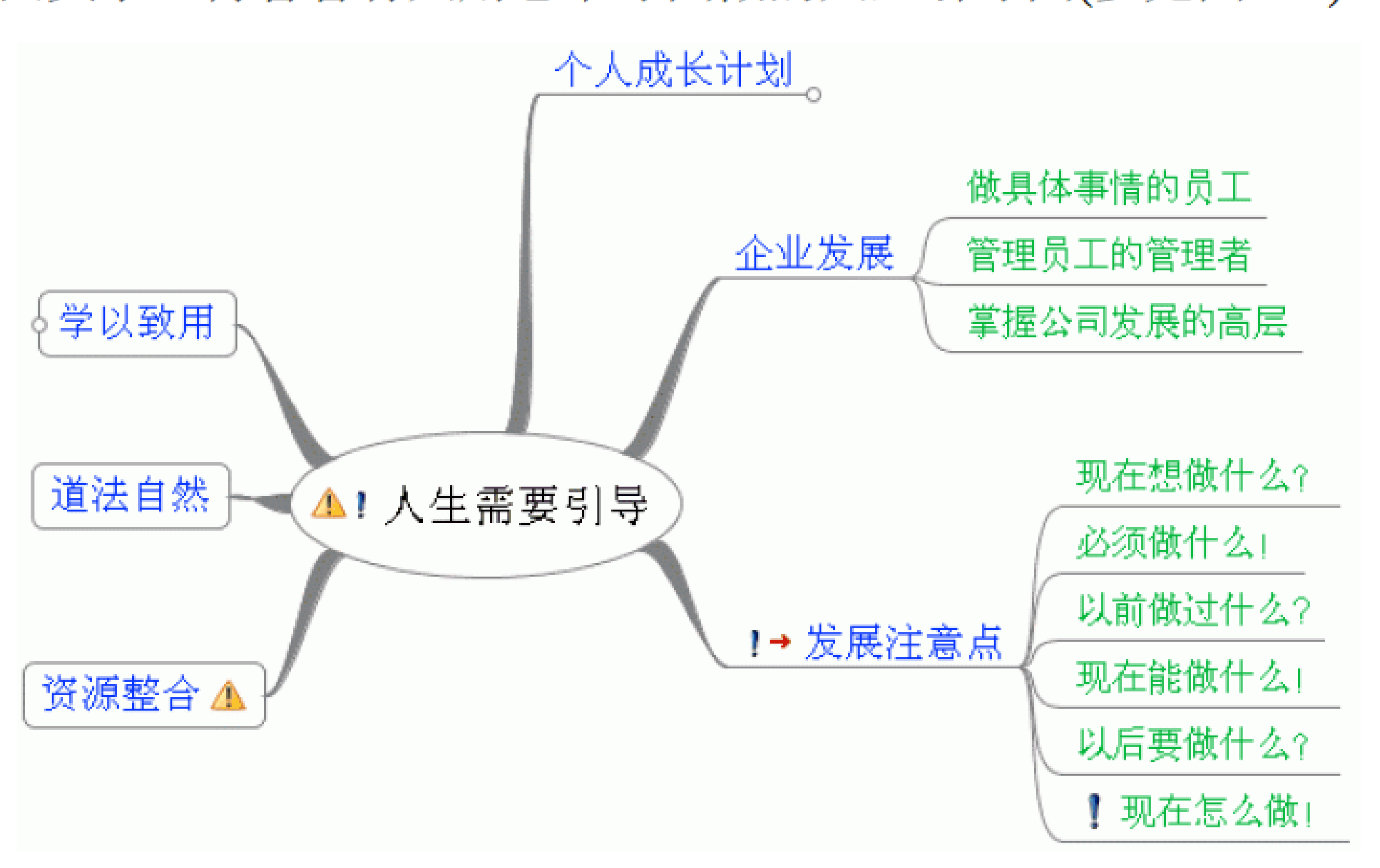


图 2-1　用思维导图做的人生引导图

1. 好处和用法

基本上来说，思维导图摆脱了呆板的线性思考，唤醒创造力，使做笔记成了一种乐趣。我们能用思维导图做什么呢？

记笔记、头脑风暴(个人或团队)、解决问题、学习和记忆、规划、研究和强化众多源头的信息、展现信息、洞察复杂对象、唤醒你的创造力，还有更多……

很难准确地说出思维导图用法的数量，但思维导图的确有助于理清思考任何事情、不同情景的思路：个人、家庭、教育或事业。计划每天的生活或者规划一生、做读书笔记、启动项目、规划和撰写报告书，撰写博客。

也就是说，思维导图适用于任何事情。

2. 思维导图的画法

思维导图的画法十分简单：

(1) 首先在一个空白纸的中间地方把你要发散的概念写出来或画出来，建议将纸张横向使用。

(2) 围绕这个中心主题发散相关副主题，并将每个副主题和中心主题用一条线连接起来。

(3) 重复同样的过程，不断逐级产生副主题，并将这些与对应的副主题连接起来。

一些建议：

- 大量使用颜色、图形和符号。尽可能可视化出来，大脑会感谢你。很多不愿尝试的人，借口说他们不是“艺术家”，别让这个借口成为不去尝试的理由。
- 主题标签尽可能简短，用一个字或者就用一个图片。特别是当你第一次使用思维导图的时候，总是很想用一个完整的短语。不过不断找机会缩短成单词或一个图形，那样你的思维导图将会更为有效。
- 改变文字的大小、颜色和排列方式，改变线条的粗细和长度等。尽你所能，多用可视化元素来强调重要的地方。每一个小地方都会激励大脑。

3. 总结

思维导图绝对是一种令人着迷并且丰富的主题，这里仅仅说了点皮毛而已。如果想获取更多的参考材料，可使用 Google 进行搜索。

目前常用的软件为 FreeMind 和 MindManager，前者为开源软件，后者为商用软件。

2.3　FreeMind——梳理你的思路

又是一堆的事情，真头大；哪件事得先做、哪件可后做？一点头绪也没有。

工作、还是工作，一堆堆的计划，怎么安排？当你感到思维杂乱、头绪繁多时，可以试试 FreeMind。使用 FreeMind，将你想到的全部记下来，不管是面面俱到的观点还是瀑布式的构思，先记下来；然后再去整理，看看各个点之间是否有关联或重复。整理整理，思路就清楚了。

比如：你要选择某一件事情，可以先把可选择的对象列出来，依次分析每个对象的优劣，然后再和自己的现状及计划进行对比，很快就能确定你的选择。

FreeMind 是一套用 Java 编写的开源思维导图/心智(MindMap)软件，可以帮助你整理思路，通过将思路图形化、结构化(也就是将思维的每一个环节用图形表示)，使一些随机的内容之间建立起有机的联系。

❖ **注意：**

浩为已修改 FreeMind 的源码，自带 Java 运行环境，解压后运行 0setup.bat 即可使用。已修改 Freemind.exe，启动时先检查是否自带 JRE(Java 运行环境的简称)。唯一不足的是，修改版在 Windows Vista 下无法关联文件，即不能直接双击文件(文件类型为 mm，即后缀名为 mm)来打开。下载地址为 http://zyt.howwe.net/me.php?564。

FreeMind 还可以用于：

- 日程管理
- 管理项目(包括子任务的管理及状态、时间记录、资源链接管理)
- 笔记或知识库
- 文章写作或头脑风暴
- 文件及文件夹管理

FreeMind 还可以用作系统构架图。

使用说明：FreeMind 的使用特别简单，基本功能使用快捷键即可完成，在此不再介绍。唯一要说明的是两个节点之间连接的删除方法，在要删除的连线上右击鼠标，从弹出的菜单中选择**删除页内(箭头)连接**即可。

两个最常用的快捷键：

- 回车：插入新的平行节点，即在当前节点的父节点下新建节点。
- Insert：插入新的子节点，即在当前节点下新建节点。

其他说明：要使节点变为泡框，可选择父节点，再选择格式里的泡框，则父节点及父节点下的所有子节点全部变为泡框。

2.4 FreeMind 应用范例

1. 日常工作目录

D:\howwe 为 hocat 的上一级目录，可作为工作目录的范例(前面有说明，再次列出是为了显示其重要性)，如图 2-2 所示。

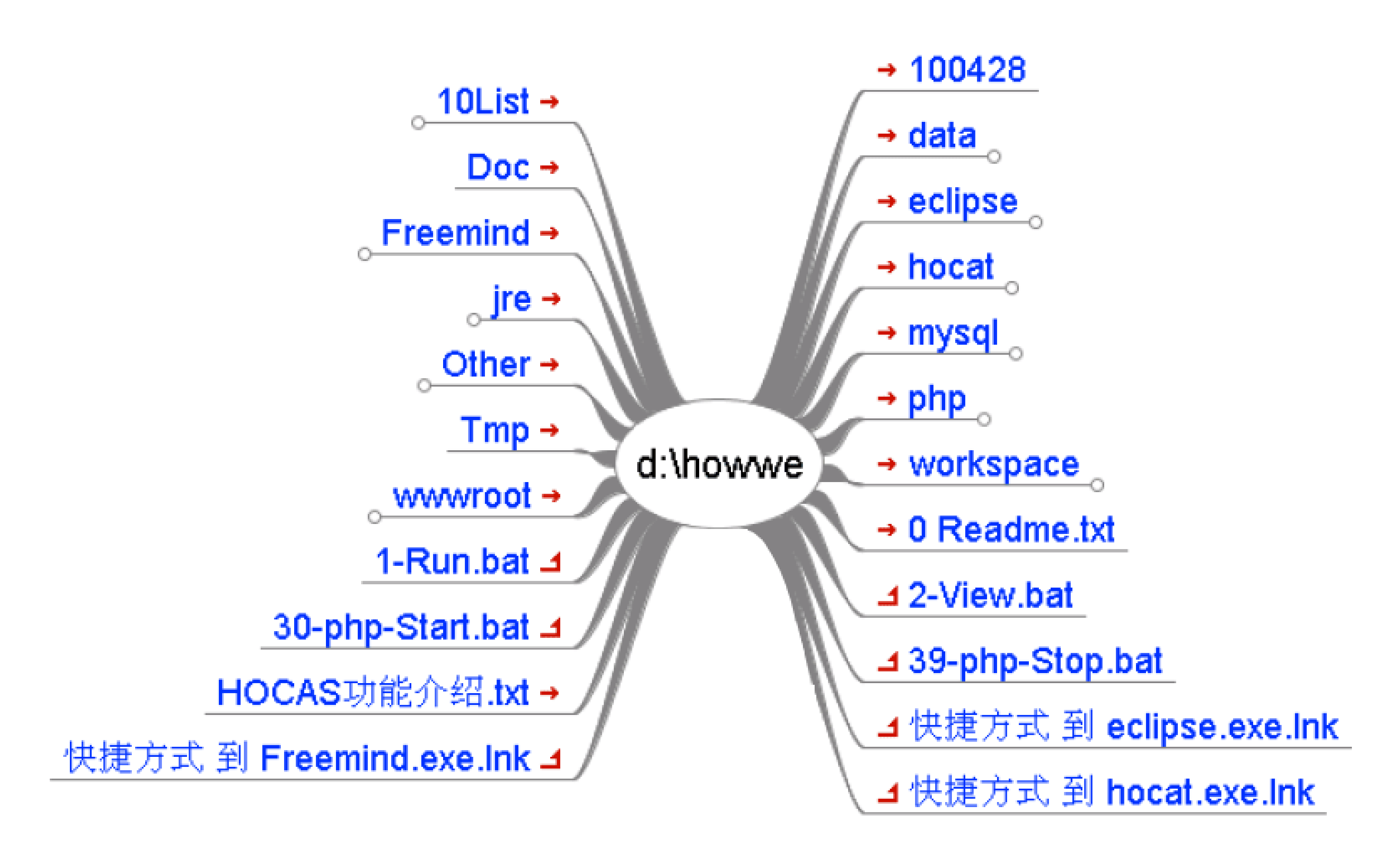


图 2-2　howwe 目录的结构

图 2-2 使用 FreeMind 生成，这是 FreeMind 的一项功能：将目录导入。

使用这一功能，可以更直观地显示目录及文件的结构，同时可以对目录或文件添加说明。如果要打开文件或目录，请移动鼠标至文件或目录前的图标上，待鼠标变为手形，即可打开。

再次说明：尽量不要在 FreeMind 文件中删除文件或目录，因为有时可能会误删除，即在不知不觉中文件就可能没了。若想修改，最好将 mm 文件保存到其他的目录，即 mm 文件不能和导入的目录在同一目录或父目录中。

在日常工作中使用这种结构的目录，将有助于工作的顺利开展，形成一种良好的工作习惯。比如日常文件的管理，将有助于养成按日期存放及整理的习惯，以后要找文件时，便不会找不着。

2. 日程管理(见图 2-3)

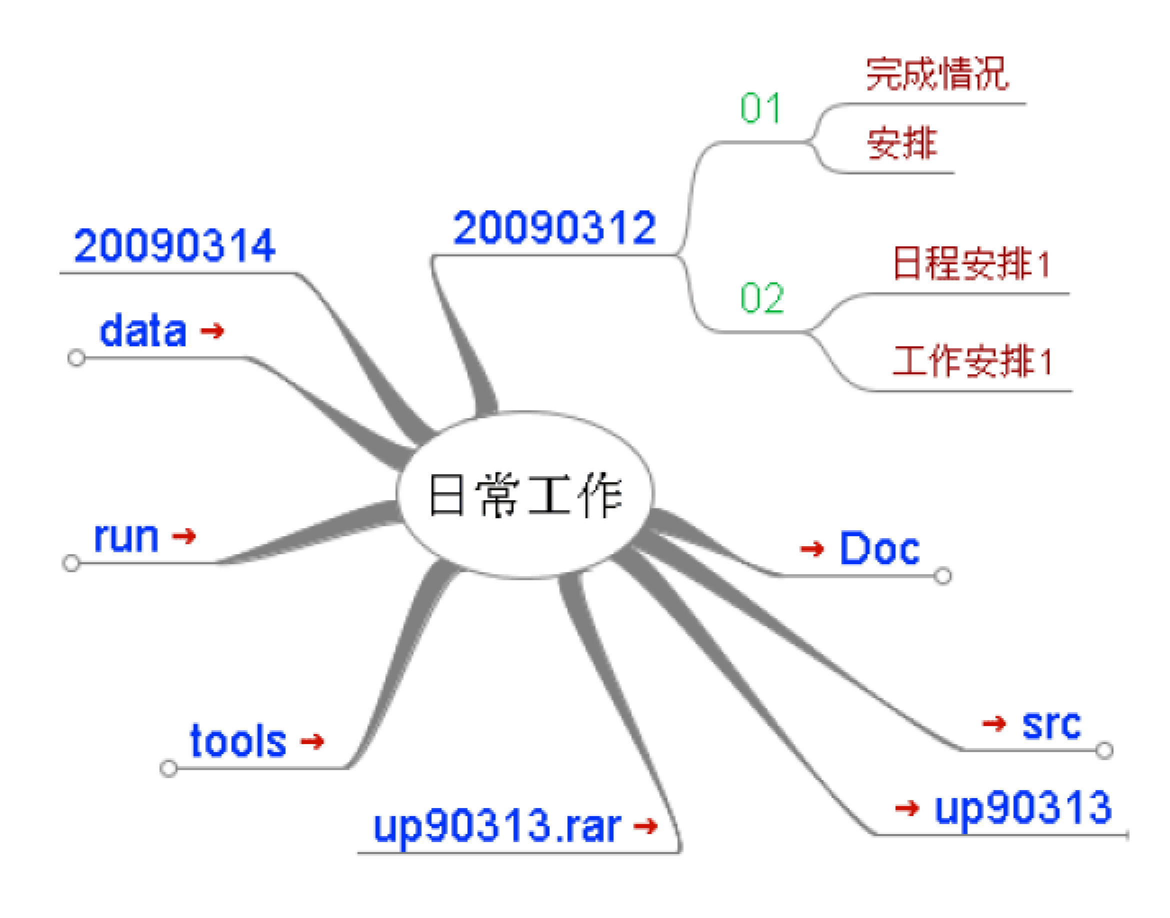


图 2-3　日常工作范例

在图 2-3 所示的日程管理范例中，查看 20090312 节点，可以事前安排日程、工作，并事后进行跟踪。已导入如下几个目录：20090314、Doc、data、src 等，可以直观便捷地查看目录及文件。

3. 论坛分析

听说某乡亲论坛窝里斗，以致分裂。论坛到底如何定位？尽管没有直接使用 FreeMind 来分析，但采用了思维导图的思想。以下是其中的几点建议：

- 论坛应该在整合资源方面为会员提供方便。这个社会很简单，需要抱团，个人的能力再强，也是有限的。
- 网站的定位，总的来说，就是要让上这个网站的人能得到他想要的。
- 人群的分类，可分为以下几类：
  - 上班族可用来消遣时间。
  - 新人期望能尽快融入社会。
  - 大龄或适龄男女希望能找到另一半。
  - 企业或个人希望整合整合资源。
  - 学生期待能对其有所帮助，在外地读书的人很多，更需要老乡的指导或帮助。
  - 热心人希望能为别人排忧解难。
- 应该建立一个面向全国的乡亲论坛，而不是一堆的地方性分论坛。

4. 辅助考研

有个将在 2011 年考研究生的大三学生问我，“考研复习，每天的各个时间段和对应要完成的任务可不可以做成思维导图？”。我简单说了一下，他就明白了，试着画了一张。先把大体轮廓做好了，然后往后细分抓出重点，越详细越好。拿出点时间来制订计划，肯定比盲目往下学有好处。不仅是考研，以后的工作、生活更是如此。

2.5 学用 Google 快速查找知识

当今社会，知识呈爆炸式增长，即使你一刻不停地学习，也无济于事，怎么办？学以致用，当你要什么，就去查找，怎么查？

查找知识，建议使用 google.cn(现将跳转到 google.com.hk)的高级搜索，如图 2-4 所示。地址为 http://www.google.cn/advanced_search?hl=zh-CN。



图 2-4　Google 高级搜索

选择 **100 项结果**，在**包含全部字词**处输入内容，这样可以快速查找自己所要的信息。如果显示的搜索结果太多，可在输入内容的地方添加限制。

例如：查找**思维导图**，发现搜索结果特多；可将搜索结果限制为**软件**的记录，输入**思维导图 软件**，注意中间用空格隔开，则很快就能查到需要的内容。

为啥要选 100 项？因为选 100 项查找更方便。如果采用默认的 10 项，当你看完 10 项结果后，如果没有你想要找的内容，就得看第 2 页、第 3 页，甚至第 10 页。如果选择显示 100 项呢，一页就有 100 项结果，再说还可以对这 100 项结果进行对比，到底哪项是你要找的内容。

依我的经验，一般大多在第 1 页就能查到想要的东西，当然也有不少要到第 2 页，甚至更多页才能找到。

建议将某次选择 100 项后的查询作为一个书签，这样下次打开时就不用再选了。

第 3 章

Hello World 范例

我们先从最简单的代码入手，输出一个字符串“Hello World”，而字符串就是一种数据类型。而后再输出多个，之所以能输出多个，是因为我们定义了变量并使用了控制结构。循环结构就是控制结构的一种。

变量之所以能定义，也是因为程序语言定义了数据类型。绝大部分程序都用来处理数据，处理数据时，处理的对象就是已结构化的多种数据类型的组合体，这个组合体一般称为数据结构。所以程序语言的三大基本知识点分别是：数据类型、控制结构、数据结构。

有人说一个稍微有点编程背景的普通人，只需要学习 PHP 半天时间，就可以上手开始开发 Web 应用；甚至一个从没有编程经历只做 Photoshop 的人，学两天 PHP，就能到处接活给人家开发网站，一个人全部搞定。可见 PHP 的简单，然而 PHP 5 提出的面向对象等众多概念却使人很迷茫。在此特别提醒，编程语言只是用来为你解决问题的，你能用它解决问题即可，其他的不用浪费时间，简单来说就是你得抱着“学以致用”的观点去学，而不是盲目追求最新技术。

3.1 输出单一字符串

先新建一个最简单的程序——03hello.php。

如图 3-1 所示，在 me 项目上单击鼠标右键，然后选择 **New** => **PHP File** (中文版对应为 **新建** => **PHP 文件**)。如图 3-2 所示，在 **File Name** 处键入 03hello.php 作为文件名称，然后单击 **Finish**。

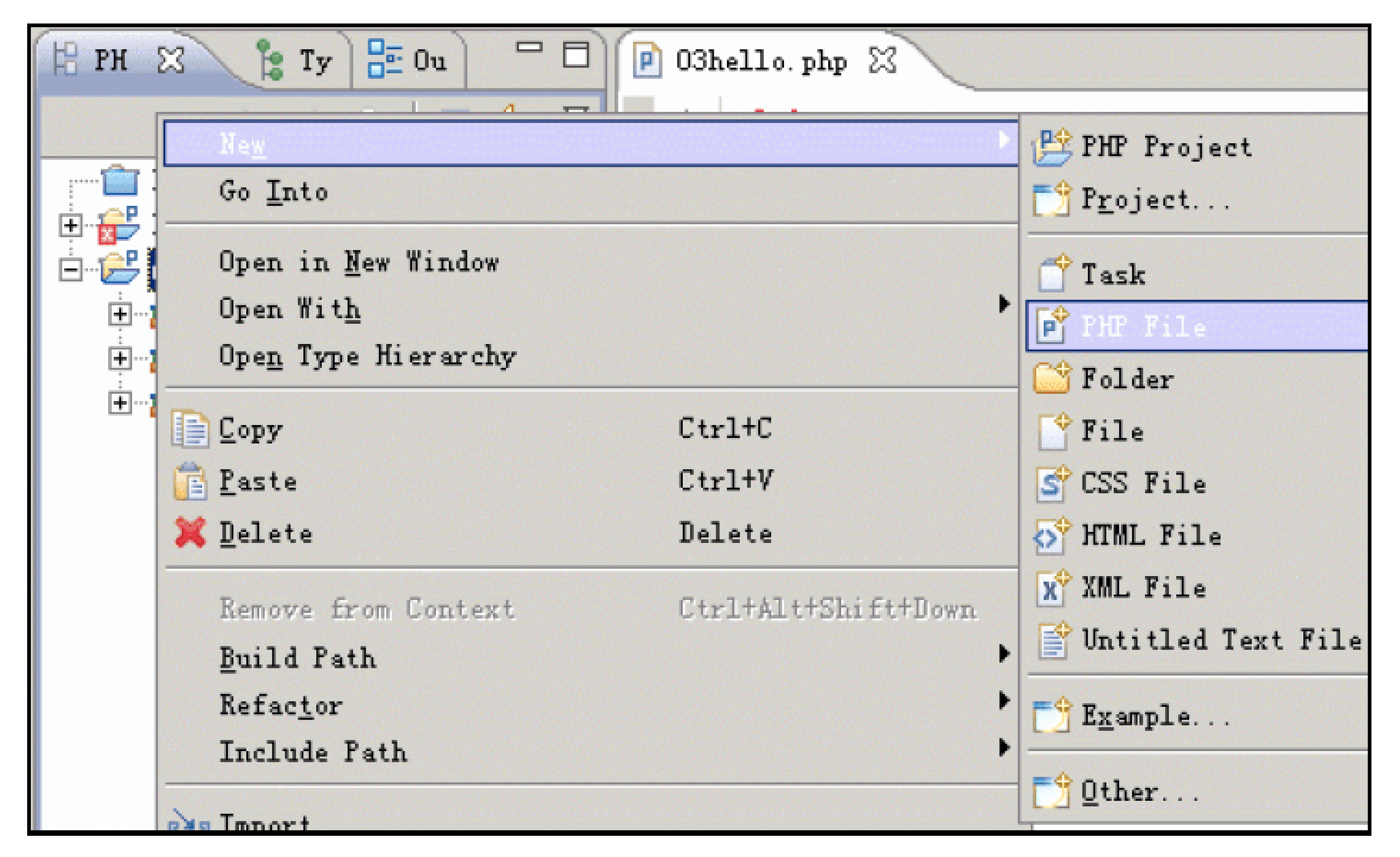


图 3-1　创建 PHP 文件

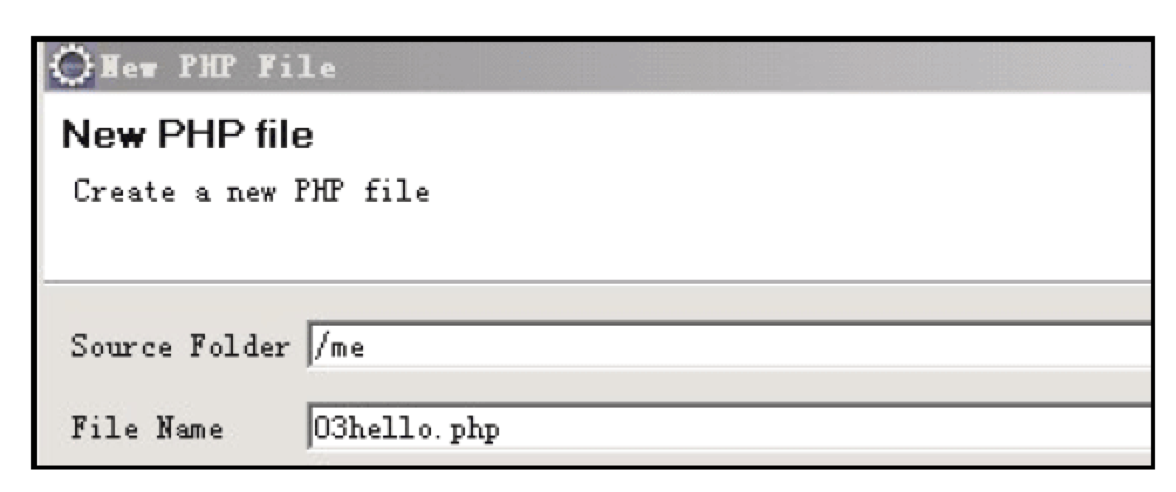


图 3-2　创建 03hello.php 文件

这样将在编辑器区域创建一个包含“<?php” 的 PHP 文件。另起一行添加如下代码，保存后如图 3-3 所示。

```
//输出单个Hello World
echo 'Hello World!';
?>
```

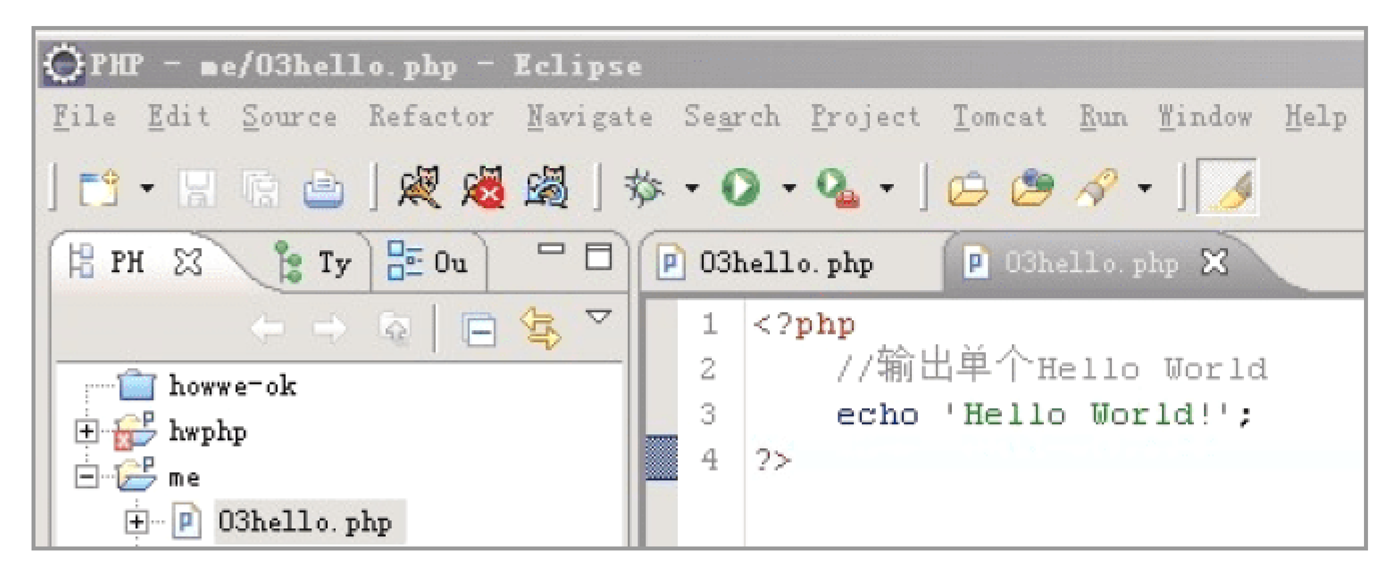


图 3-3　编辑器中的 Hello 类

❖ 提示：

PHP 有三种注释形式——C、C++所使用的 “/*...*/” 和 “//”，以及 Perl 的 “#”。

在键入时你将见到 Eclipse 编辑器的一项特性：代码自动完成。当键入括号或双引号时，Eclipse 会自动提供配对的符号，并将光标置于符号对之内。

在其他情况下，可以通过按 Ctrl+Space 组合键来调用代码自动完成功能。代码自动完成功能提供了上下文敏感的建议列表，你可以通过键盘或鼠标从列表中选择。这些建议可以是针对某个特定对象的方法列表，也可以是基于不同关键字(比如 for 或 while)展开的代码片段。

说明：echo 关键字用于打印内容，然后终止。

通过 PHP 来输出文本的基础指令有两种：echo 和 print。在上面的例子中，我们使用了 echo 语句来输出文本“Hello World！”。

更多内容也可以用 Google 的高级搜索功能来查看，地址如下：

```
http://www.google.com.hk/advanced_search?hl=zh-CN
```

代码 `echo 'Hello World!';`中的`'Hello World!'`是一个字符串常量。什么是字符串？字符串是一种数据类型。什么是常量？值不变的变量称为常量。习惯上将常量的名称统统大写。

详细说明：

PHP 脚本块以<?php 开始，以?>结束，可把 PHP 脚本块放置在文档中的任何位置。在支持简写的服务器上，可以使用<?和?>来开始和结束脚本块。不过，为了达到最好的兼容性，推荐使用标准形式(即以<?php 开头)，而不是简写形式。

PHP 文件通常包含 HTML 标签，结构类似 HTML 文件，包含 PHP 代码；也可以不含 HTML 标签，仅有 PHP 代码，直接用 PHP 代码生成 HTML 标签。PHP 中的每个代码行都必须以分号结束。分号是一种分隔符，用于把指令集区分开来。

下面的代码中包含有 HTML 标签，关于 HTML 标签的详细介绍请浏览 **9.2 HTML 语法。**

例 0302.php：

```
<?php
    //输出单个 Hello World
echo 'Hello World!';
?>

<!--带 html 标签-->
<html>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Hello World</title>
<body>

<?php
//由于 PHP 文件的格式为 uft-8，包含汉字时，必须用@header 指定编码格式为 utf-8，否则汉
  字显示为乱码。用 meta 来指定编码也能起到相同的作用，语句如下：
//<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
@header("content-Type: text/html; charset=utf-8");
```

```
echo "<br>Hello World!";
echo "<br><br>带 html 标签";
?>

</body>
</html>
```

小知识：PHP 注释详解

PHP 支持 C、C++和 UNIX Shell 风格(Perl 风格)的注释，比如：

```
<?php
   echo "This is a test"; // This is a one-line c++ style comment
   /* This is a multiline comment
      yet another line of comment */
   echo "This is yet another test";
   echo 'One Final Test'; # This is a one-line shell-style comment
?>
```

单行注释仅仅注释到行末或当前的 PHP 代码块，`// ... ?>`或`# ... ?>`之后的 HTML 代码将被显示出来，即`?>` 跳出了 PHP 模式并返回了 HTML 模式，`//`或`#`并不能影响到这一点。

❖ 注意：

C 风格的注释在碰到第一个`*/`时就结束，要确保不嵌套 C 风格的注释。当试图注释掉一大块代码时，会很容易出现该错误，例如：

```
<?php
 /*
   echo "This is a test"; /* This comment will cause a problem */
 */
?>
```

3.2　执 行 代 码

一旦代码无错误地编译完成，你就能够从 Eclipse 菜单上选择 **Run** 来执行该程序了。有 4 种执行方式可供选择：

- 从菜单 **Run** 选择 **Run As**。
- 单击图 3-4 所示的按钮，选择 **Run As**。
- 在文件编辑区，单击鼠标右键，如图 3-5 所示，选择 **Run As**。

- 在浏览器中输入 http://127.0.0.1/03hello.php 并回车(注意：这种方式将直接在浏览器中显示结果)。

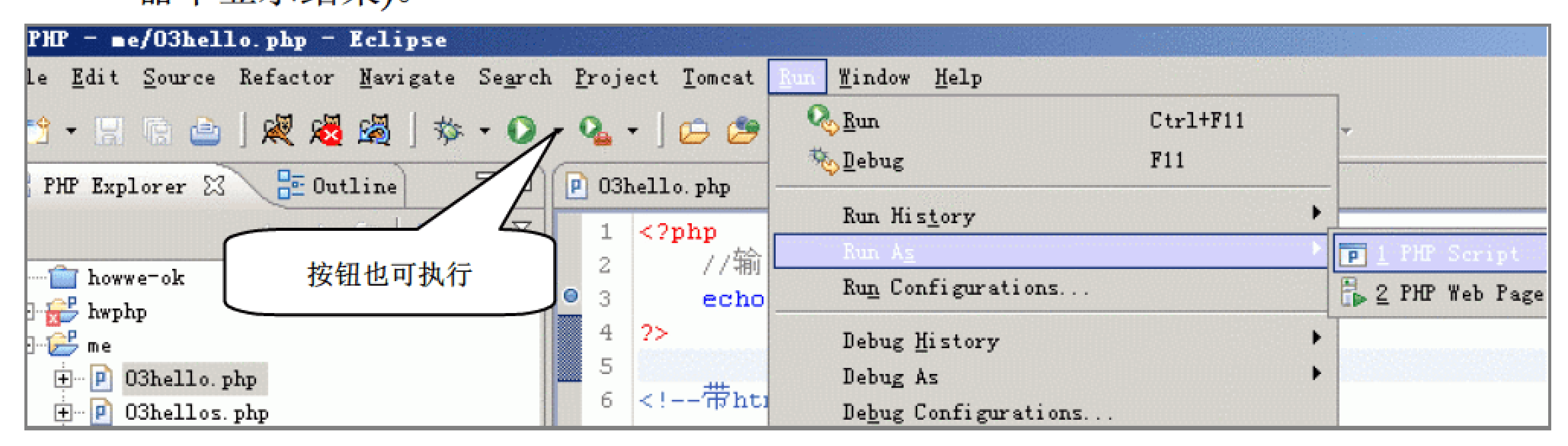


图 3-4　通过菜单或按钮执行 03hello.php

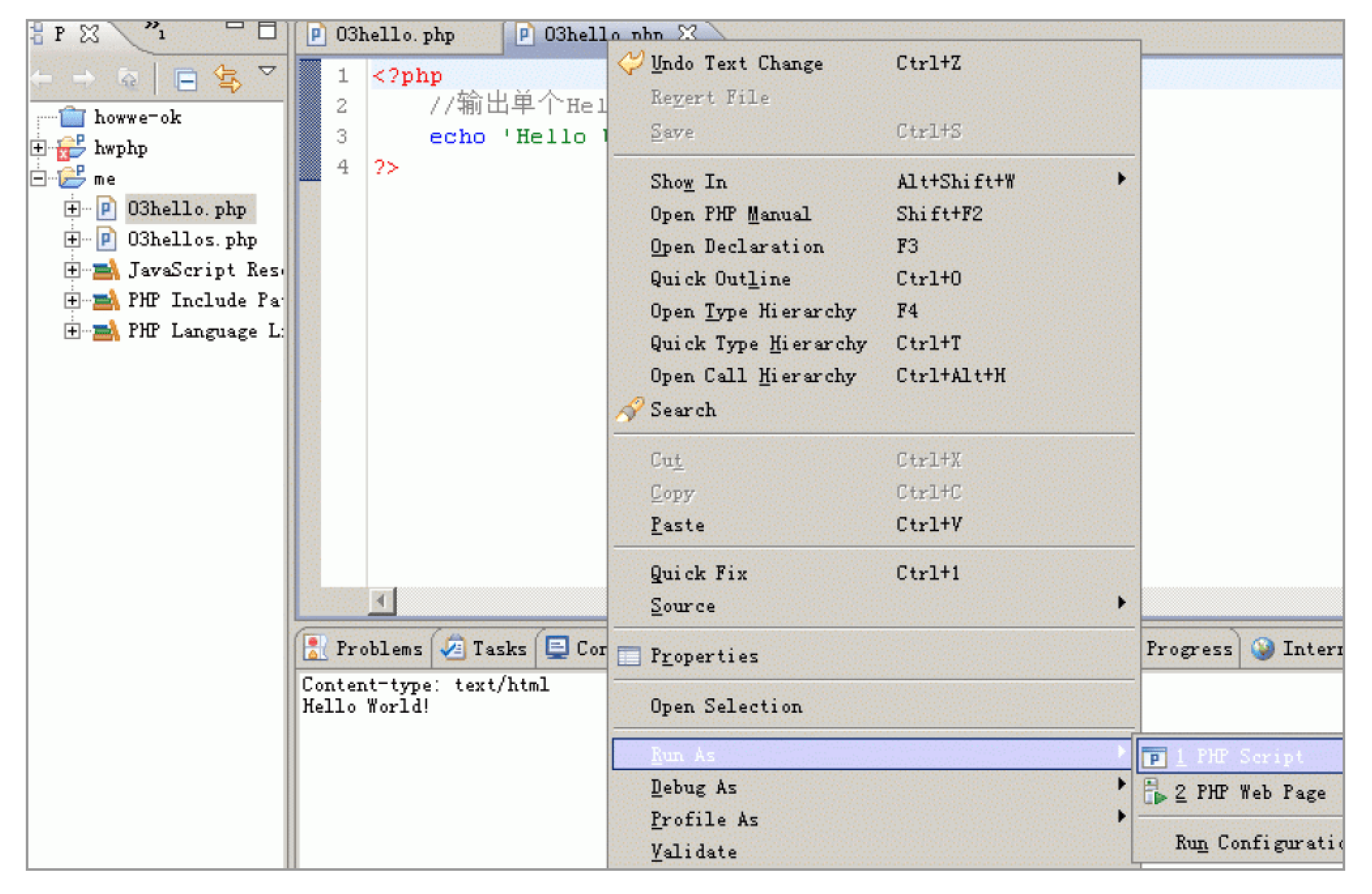


图 3-5　右键执行 03hello.php

运行后，控制台区域将出现一个 Console 标签，一个 Debug Output 标签，以及一个 Browser Output 标签显示了程序的提示及输出内容，分别如图 3-6、图 3-7 和图 3-8 所示。

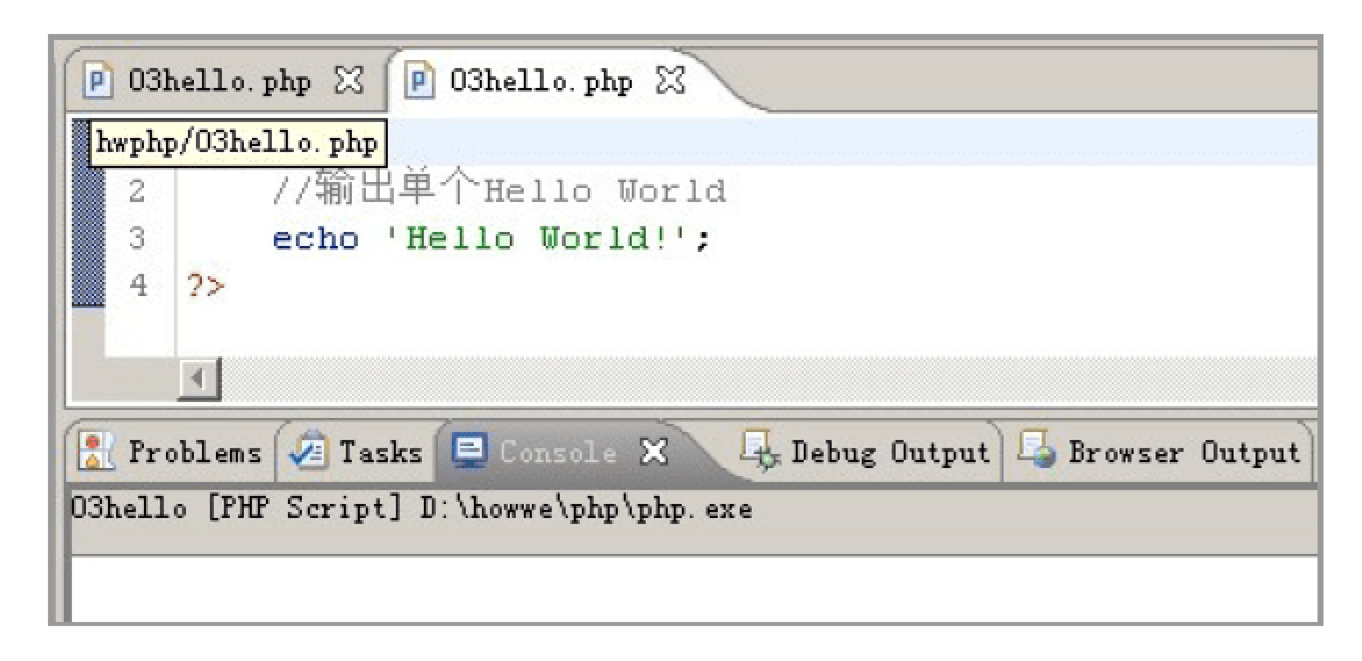


图 3-6　执行 03hello.php 后的提示

图 3-6 显示没有提示；如果有提示，则单击相应的地方，就能定位到相应的行。

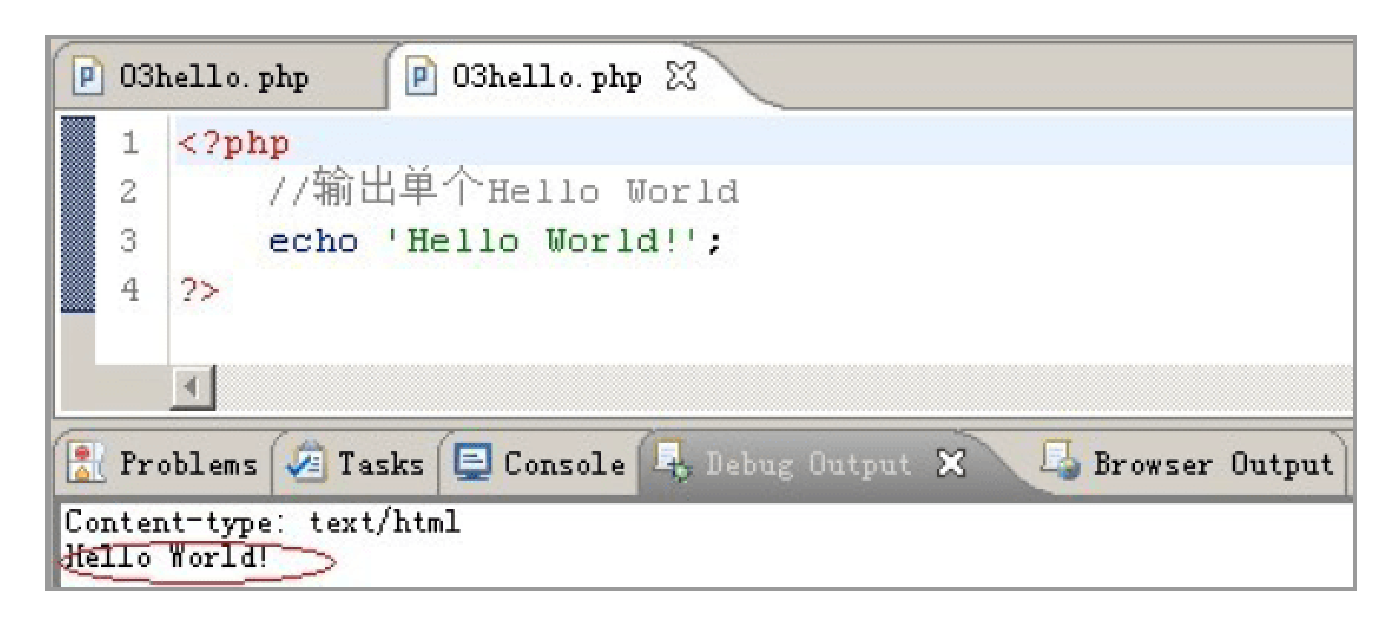


图 3-7　执行 03hello.php 后 Dubug Output 中的提示

图 3-8　执行 03hello.php 后 Browser Output 中的提示

运行结果如图 3-9 所示。

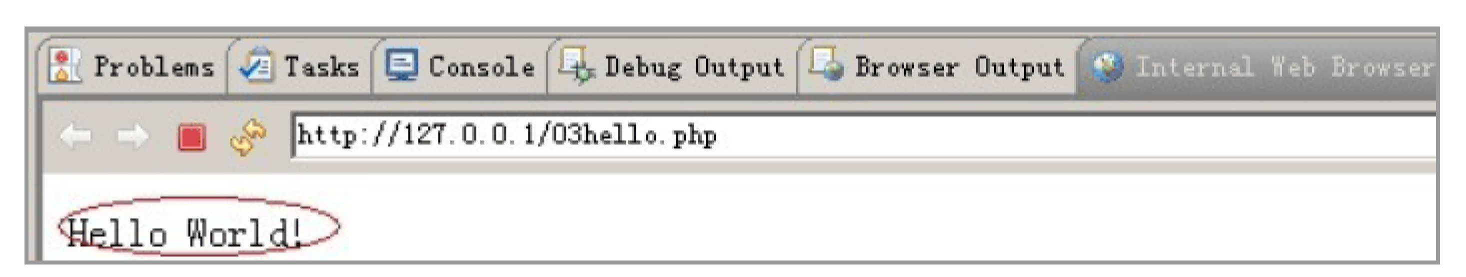


图 3-9　浏览器显示结果

3.3　输出多个字符串

在 me 项目上单击鼠标右键，然后选择 **New => PHP File** (中文版对应为 **新建 => PHP 文件**)。在 **File Name** 处键入 03hellos.php 作为文件名称，然后单击 **Finish** 。

这样将在编辑器区域创建一个名为 03hellos.php 的 PHP 文件。编辑如下代码，保存后如图 3-10 所示(注意其中变量 i 的声明是有意省略了的)。

```
<?php
//输出单个 Hello World
echo 'Hello World!<br>';
echo '----<br>';
```

```
//输出多个 Hello World
for(i = 0;i < 4;i++){
    echo 'Hello World!<br>';
}
?>
```

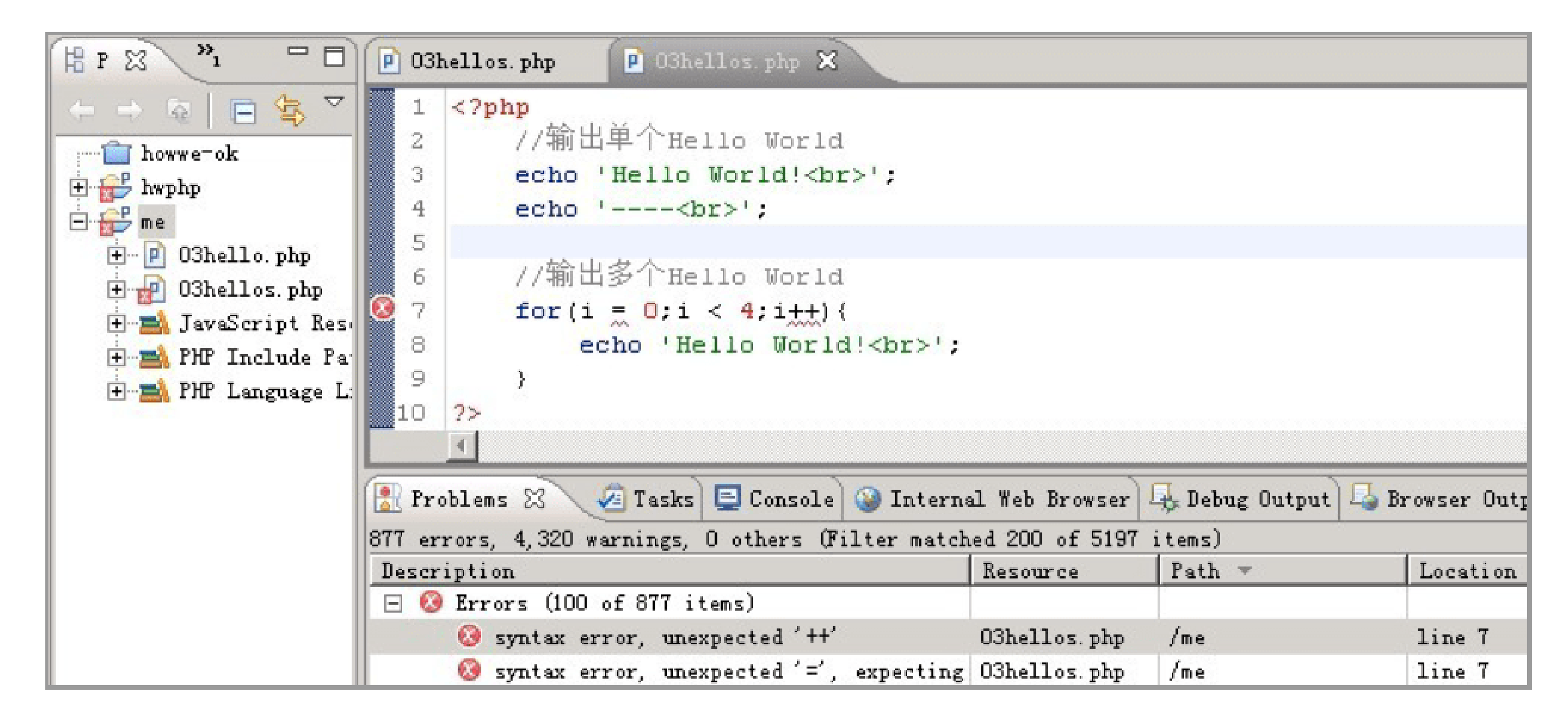


图 3-10　编辑器中的 03hellos.php

保存代码后，就能看到 Eclipse 编辑器的一项特性：语法检查。

语法检查依赖增量编译。每当保存代码后，它就在后台接受编译和语法检查。默认情况下，语法错误将以红色下划线显示，一个带白色“X”的红点将出现在左边沿。其他错误在编辑器的左边沿通过灯泡状的图标来指示；这些就是编辑器或许能为你修复的问题——即所谓的 Quick Fix(快速修复)特性。

图 3-10 所示的错误是“syntax error”，即语法错误，但没有提示“i”的定义出错，所以没有对其进行修正。

图 3-11　Quick Fix 建议

在 i 前面双击图 3-11 所示的建议，即可将建议代码插入到代码中的恰当位置，保存后按前面所讲的方法执行程序，执行结果如图 3-12 所示，一共输出了 5 个“Hello World”字符串。

03hellos.php　03hellos.php

```
<?php
    //输出单个Hello World
    echo 'Hello World!<br>';
    echo '----<br>';

    //输出多个Hello World
    for($i = 0;$i < 4;$i++){
        echo 'Hello World!<br>';
    }
?>
```

Problems　Tasks　Console　Debug Output　Browser Output

Hello World!

Hello World!
Hello World!
Hello World!
Hello World!

图 3-12　执行 03hellos.php

代码`$i = 0;`其实就是定义了一个变量。什么是变量，它与常量有什么区别？以下是两者的简单对比，详情请参考“**第 4 章　数据类型**”。

常量与变量的区别：

常量是一个简单值的标识符(名称)。正如其名称所暗示的，在脚本执行期间该值不能改变(“魔术常量”除外，它们其实不是常量)。常量默认大小写敏感，通常常量标识符总是大写的。变量用一个美元符号后面跟变量名来表示，变量名也是区分大小写的。

变量的类型通常不是由程序员设定的，确切地说，是由 PHP 根据该变量使用的上下文在运行时决定的。

任何 PHP 标签都遵循同样的命名规则。合法的标签以字母或下划线开始，后面跟着任何字母、数字或下划线。

用正则表达式可描述为：`'[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*'`。

❖ **注意：**

1) 此处所说的字母是 a-z、A-Z，以及从 127 到 255(0x7f-0xff)的 ASCII 字符。

2) $this 是一个特殊的变量，它不能被赋值。

在类 D10302Hello 中，先执行语句：

```
//输出单个Hello World
echo 'Hello World!<br>';
echo '----<br>';
```

再执行代码段：

```
//输出多个Hello World
for($i = 0;$i < 4;$i++){
```

```
        echo 'Hello World!<br>';
    }
```

这两者按顺序执行，从控制结构上来说是一种顺序结构。但后面的代码段循环执行了 4 次，属于循环结构。控制结构的详细内容请浏览“**第 5 章 控制结构**”。

3.4 调 试 代 码

可以在 PHP 调试器中运行程序。在第二个 echo 'Hello World!
'处，双击编辑器视图左端的灰色边沿，即可在此处设置一个断点。一个蓝色的点将会出现在那里，如图 3-13 所示。

图 3-13 设置断点

然后从 **Run** 菜单中选择 **Debug**。参照 3.2 节的描述，这里只是用 Debug 代替 Run，再选择 **PHP Script** 之后，透视图将自动切换到 **Debug** 透视图，其中具有许多有趣的新视图，如图 3-14 所示。

图 3-14 Debug 界面

左上角的窗格包含许多选项卡式的视图，包括 Variables、Breakpoints。单击 Variables 标签，以便我们能够看到变量$i 当前的值，如图 3-15 所示。

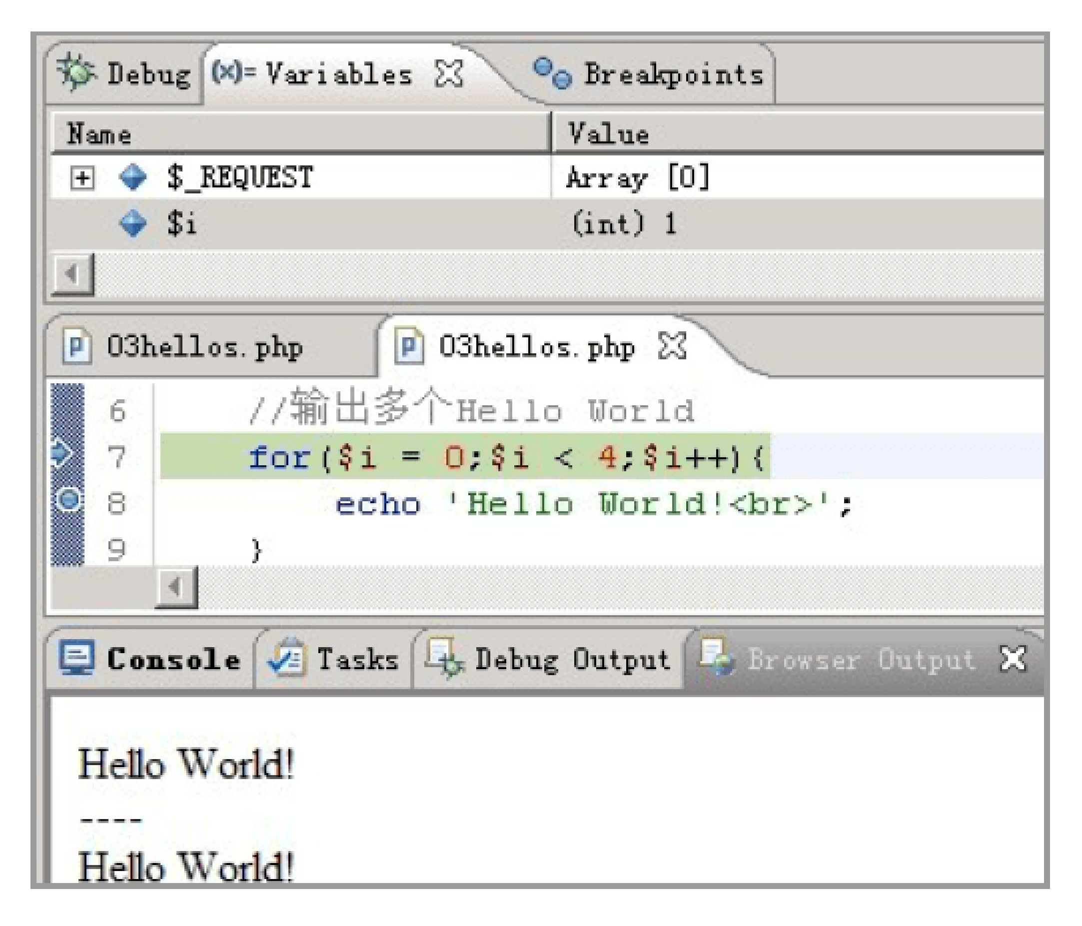


图 3-15　Variables 视图

该透视图左上角的 **Debug** 视图用来显示调用堆栈，并且标题栏中有一个工具栏，它允许您控制程序的执行，包括继续、挂起或终止，跟踪、单步执行下一条语句，或者从方法返回。

快捷键说明：

F6　单步执行语言。

F8　执行到下一个断点或结束程序(没到下一个断点就结束)，也称 Resume。

3.5　IT 培训潜规则，培训更需要实干

1. 培训陷阱

1) 培训陷阱一：名企联合招聘/招聘程序员或实习生

案例 3-1：小陈参加了国内某知名软件公司的招聘，结果面试地点在某培训机构内。刚开始有个像模像样的面试环节，然后就会有面试官(其实是培训机构的咨询师)对小陈一顿打击，说小陈这不行，那也不行，想工作就得先参加培训。想去名企？先交钱培训！(其实培训之后，也未必去得了名企。)

案例 3-2：小强刚大学毕业，想找工作，在人才市场发现某公司招聘程序员。于是上门应聘，上门后发现，根本不是招聘，直接问他：你能力限制不符合工作，培训下能接受不？

潜规则：很多黑心培训机构利用大学生求职心切的心理，打着招聘的幌子，实际上是在招生。只要先把学生骗上门，其余的就好说了……

2）培训陷阱二：保证就业，底薪>3000

案例 3-3：小李在某号称实力很强的培训机构花 10000 元学习，毕业后被推荐到某软件公司工作，试用期工资 3000，但一个月后无故被开除，再去找培训机构，培训机构说我们已经帮你推荐了。实际上小李的工资是培训机构发的，羊毛出在羊身上，只不过少赚点而已。

潜规则：很多没有实力的培训机构没法解决就业，怎么办？把从学生身上收取的高额学费拿出一小块给企业，企业免费或廉价地用了一个月，也没什么损失，唯一可怜的只有学生……

3）培训陷阱三：保证安置工作，100%就业

案例 3-4：小王在某知名培训机构学习 Java，该机构承诺毕业后推荐工作，于是小王多次被机构推荐到单位去面试，可就是没人接收。直到小王信心全无，很烦为止，自然小王也没能找到好工作。

潜规则：很多培训机构都用最好听的承诺把你骗进来，当然最后能不能找到工作，那就不像刚开始说的那么好了。

4）培训陷阱四：订单培养

案例 3-5：小明参加了某培训机构的学习，据说与该培训机构合作的企业有三百家，每个月都有大量企业直接来培训机构要人。学完才发现，来招人的就那么两三家，而且待遇很低。

潜规则：有些培训机构说自己合作企业近千家，还能拿出和几家企业签订的定向委培、人才合作培养等协议给你看。其实，公章这种东西是真是假你也看不出来。

5）培训陷阱五：名师授课

案例 3-6：小黑咨询某培训机构，该机构号称自己的老师做过价值几百万的项目，多少年项目经验，多少开发经验，绝对是一流师资。结果报名学习之后，发现该老师讲的也就是比大学老师好那么一点点。

潜规则：能做项目的人很多，但做过项目不代表会讲课，毕竟讲课还是需要一些教学理念。项目经理太多，也不是做过项目经理，就能讲好课。好老师一般比较少，而且要价比较高，所以很多培训机构都是找些不入流的老师滥竽充数。

案例 3-7：小白咨询某培训机构，该机构介绍时说师资阵容强大，百度一下老师名字，绝对都是业内牛人。试听过该“牛人”几次课后，小白于是报名学习，刚学习时有名师授课，上几天课后发现给自己讲课的全换成了普通老师。或者就是名师就负责讲讲就业指导和简历制作等。

潜规则：很多好一点的培训机构一般都有一两个名师，不过这些牛人都是负责招生的，招生的讲座、前面的试听课是他们讲，等学生过了退费期，就马上换个普通老师来讲课。

2. 培训需要实干

培训到底需要什么？培训要能让学生学到真本事，做事的本领，而不只是一堆知识，或者是一堆习题。

在现实社会中，即使你拿再多的证书，而动手能力一般，甚至很差，公司会要吗？公司要的是做事的人，而不是一堆证书，只有证书而没有动手能力的人，对公司来说无异于废物。

其实，如果真想自己以后工作顺利一点，就得培养动手能力，找培训就得找一家能培养动手能力的公司。

到底要学什么呢，可以看看**"9.7 学什么，学以致用"**。

3. IT 技能层次

精通：能够掌握此技术 85%以上的技术要点，使用此技术时间超过两年，并使用此技术成功实施过 5 个以上的项目。能使用此技术优化性能或代码，做到最大可能的重用。

熟练：能够掌握此技术 60%以上的技术要点，使用此技术时间超过一年，并使用此技术成功实施过 3 个以上的项目。能使用此技术实现软件需求，并有经验的积累，在实现之前能做优化设计，尽可能实现模块或代码的重用。

熟悉：能够掌握此技术 50%以上的技术要点，使用此技术时间超过半年，并使用此技术成功实施过 1 个以上的项目。能使用此技术实现软件需求。

了解：可以在实际需要时，参考技术文档或帮助文件，满足你的需要，基本知道此项技术在应用中所起的作用，能够调用或者使用文档提供的接口。

打个比方，吃饭吃了几十年，可绝大多数人还是没法说他已经"精通"吃饭了，最多只能是"熟悉"。为啥？吃饭是为了生存，大多数人吃饭只是满足了生存的基本需要，即吃饭只处于"了解"阶段。吃饭的"熟悉"阶段，得满足生存的需求，即要通过吃能使自己少生病。吃饭的"熟练"阶段，需对生存做优化，即通过吃，能使自己身心健康。而吃饭的"精通"阶段，在于对生存做最大优化，即通过吃，能使自己延年益寿。

第 4 章

数 据 类 型

IT 知识点数量很多，即使你一刻不停地学，一辈子也学不完。那该怎么呢？万变不离其宗，每门学科都有自己的基本原理。了解基本原理，其他东西就可以举一反三，从而达到事半功倍的效果。

我们开发项目，整体思想是第一位，开发语言的基本知识是第二位。该掌握哪些基础知识呢？数据类型、数据结构、控制结构是基础的基础。

数据类型是编程的第一步，到底什么是数据类型？数据类型是用来约束数据的解释。数据类型描述了数值的表示法、解释和结构，以及算法操作，或是对象在内存中的存储区，抑或是其他存储装置。

程序的目的是什么？简单来说就是运算，除了运算还是运算。有以下几种运算：算术、逻辑、比较、递增、递减。

PHP 变量的类型通常不由程序员设定，而是由 PHP 根据该变量使用的上下文环境在运行时决定。

4.1 计算机中的数据类型

计算机中的所有数据，最基本的单位是比特(bit)，也叫位，即一个二进制数据，用 0 或 1 表示。数据的最小寻址单位称为字节(通常是 8 bit，以 8 个位为一组)。在同一时间，计算机能处理的二进制数的位数叫字长。32 位计算机在同一时间能处理字长为 32 位的二进制数据。64 位计算技术从 2004 年推出至今，目前还没有得到大规模应用。

字节是计算机信息技术用于计量存储容量和传输容量的一种计量单位，1 个字节等于 8 位二进制数。字节的英文名称是 Byte，是 Binary Term 的缩写。Byte(字节)可被缩写成 B，例如 MB 表示 MegaByte。

此类单位的换算为(参见图 4-1)：

1 bit　= 1 位/比特(二进制数据)
1 Byte = 8bit(字节)(Byte，Binary Term)
1 字母 = 1Byte = 8 bit
1 汉字 = 2Byte = 16 bit
1 KB　= 1024 Byte(2的 10 次方字节)(KB，KiloByte)
1 MB　= 1024 KB (2的 20 次方字节)(MB，MegaByte)
1 GB　= 1024 MB (2的 30 次方字节)(GB，GibaByte)
1 TB　= 1024 GB (2的 40 次方字节)(TB，TeraByte)



图 4-1　计量单位的换算

4.2　基本数据类型

在编程语言中，常见的数据类型为原始类型，PHP 主要有 4 种基本数据类型：布尔型(boolean)、整型(integer)、浮点型(float)、字符串(string)。但变量的类型通常不是由程序员决定，而是由 PHP 运行过程决定。如果想指定类型，可以使用 cast 或函数 settype()。

程序员可以利用多种数据类型：某些由编程语言定义，某些由外部库定义，还有些是由程序员定义。编程语言提供若干原始数据类型，作为程序及专用复合类型的建立基础。很多编程语言都依赖于特定的计算机类型和对数据类型属性的具体编译实现，但 PHP 变量的类型通常不由程序员设定，而是由 PHP 根据该变量使用的上下文在运行时决定。

另外，PHP 还有两种复合类型——数组(array)和对象(object)，两种特殊类型——NULL 和资源(resource)。

1. 布尔型(boolean)

这是最简单的类型，boolean 表达了真值，可以为 TRUE 或 FALSE，不区分大小写。通常用某些运算符返回 boolean 值，并将其传递给控制流程。

例 0401.php：

```
<?php
$foo = True; // 赋值，assign the value TRUE to $foo
?>
```

例 0402.php：

```
<?php
// == 是一个操作符，它检测两个变量是否相等，并返回一个布尔值
if ($action == "show_version") {
    echo "The version is 1.23";
}

// 这样做没有必要，因为可以使用后面的简单方式：// if ($show_separators)
if ($show_separators == TRUE) {
    echo "<hr>\n";
}
?>
```

如果要将一个值转换成 boolean，可以用(bool)或(boolean)来强制进行转换。但很多情况下不需要强制转换，因为当运算符、函数或流程控制结构需要一个 boolean 参数时，该值会被自动转换。

当转换为 boolean 时，以下值为 FALSE：布尔值 FALSE 自身、整型值 0(零)、浮点型值 0.0(零)、空字符串、字符串"0"、不包括任何元素的数组、不包括任何成员变量的对象(仅 PHP 4.0 适用)、特殊类型 NULL(包括尚未设定的变量)、从没有任何标记(tags)的 XML 文档生成的 SimpleXML 对象，而其他值均为 TRUE(包括任何资源)，如－1 和其他非零值(不论正负)。

例 0403.php：

```
<?php
var_dump((bool) "");        // bool(false)
var_dump((bool) 1);         // bool(true)
var_dump((bool) -2);        // bool(true)
var_dump((bool) "foo");     // bool(true)
var_dump((bool) 2.3e5);     // bool(true)
var_dump((bool) array(12)); // bool(true)
var_dump((bool) array());   // bool(false)
var_dump((bool) "false");   // bool(true)
?>
```

2. 整型(integer)

整型的数值范围和平台有关，常用的 32 位机范围是：－2 147 483 648(－2^{31})到 2 147 483 647(2^{31}－1)，PHP 不支持无符号整数。Integer 值的字长可用常量 PHP_INT_SIZE 表示，最大值可用常量 PHP_INT_MAX 表示。

整型值可用十进制、十六进制或八进制表示，前面可加上可选的符号(－或+)。八进制

前必须加上 0(零)，十六进制前为 0x。

❖ **注意：**

如果向八进制数传递了一个非法数字(8 或 9)，则后面其余数字会被忽略。

例 0404.php：

```
<?php
$a = 1234; // 十进制数
$a = -123; // 负数
$a = 0123; // 八进制数 (等于十进制 83)
$a = 0x1A; // 十六进制数 (等于十进制 26)
var_dump(01090); // 八进制 010 = 十进制 8
?>
```

如果给定的数或运算结果超出了 integer 的范围，则将会被解释为 float。

❖ **注意：**

PHP 中没有整除运算符。1/2 产生出 float 0.5。可以用 round() 函数或 int 舍弃小数部分。

例 0405.php：

```
<?php
var_dump(25/7);          // float(3.5714285714286)
var dump((int) (25/7)); // int(3)
var_dump(round(25/7));  // float(4)
?>
```

3. 浮点型(float)

浮点型，也叫浮点数、双精度数或实数，字长和平台相关，通常最大值是 1.8e308，并具有 14 位十进制数字的精度(64 位 IEEE 格式)。可以用以下代码段中的任一语法定义：

例 0406.php：

```
<?php
$a = 1.234;
$b = 1.2e3;
$c = 7E-10;
?>
```

❖ **注意：**

不要精确地去用有限位数表达某些十进制分数，否则会造成混乱结果：十进制的 1/3

变成了 0.3；int((0.1+0.7)*10)通常会返回 7，而不是预期的 8，因为该结果内部表示类似于 7.9。

所以永远不要相信浮点数结果精确到了最后一位，也永远不要比较两个浮点数是否相等。如果确实需要更高的精度，应该使用任意精度的数学函数或 gmp 函数。

小结：整数或浮点数又统称为数值类型。

例 0407.php：

```
<?php
$a = 1234; # 十进制数
$a = -123; # 负数
$a = 0123; # 八进制数 (等于十进制数的 83)
$a = 0x12; # 十六进制数(等于十进制数的 18)
$a = 1.234; # 浮点数"双精度数"
$a = 1.2e3; # 双精度数的指数形式
?>
```

4. 字符串(string)

字符串可以由单引号或双引号定义，字符串中首个字符的位置定义为 0，而不是 1。

❖ **注意：**

被单引号引出的字符串只能定义引号内定义的内容，而双引号引出的字符串可以被扩展。在双引号字符串中可以使用反斜杠(\)，或在字符串中加入转义序列和转换字符。

例 0408.php：

```
<?php
$first = 'Hello';
$second = "World";
$full1 = "$first $second"; // 双引号输出时转义变量，输出：Hello World
$full2 = '$first $second'; // 单引号输出时不转义变量，输出：$first $second
$full3 = "01DC studio,.\" 2000 copyright.\" " ;
?>
```

请注意最后一行，如果需要在字符串中使用双引号，可以使用反斜杠字符，这里的反斜杠用来使双引号的功能改变。

也可以将字符和数字利用运算符号连接起来。

常用字符串函数：strlen()函数用于计算字符串的长度，strpos()函数用于在字符串内检索一段字符串或一个字符。

5. 数组

PHP 中的数组实际上是一个有序映射，即把 values 关联到 keys 的类型。此类型在很多方面做了优化，因此可以把它当成真正的数组，可作为列表(向量)、散列表(是映射的一种实现)、字典、集合、栈等。数组元素的值也可以是另一个数组，树型结构和多维数组也可以。

数组接受任意数量的用逗号分隔的“键(key) => 值(value)”对。

```
array(  key =>  value
     , ...
     )
// 键(key)可以是一个整数(integer)或字符串(string)
// 值(value)可以是任意类型的值
```

例 0409.php：

```
<?php
$arr = array("foo" => "bar", 12 => true);

echo $arr["foo"]; // bar
echo $arr[12];    // 1
?>
```

key 可以是 integer 或 string。如果 key 是一个 integer，则被解释为整数(例如："8" 将被解释为 8，而"08"将被解释为字符"08")。key 中的浮点数被取整为 integer，不能将数组和对象作为键(key)。

❖ **注意：**

数组的索引默认从 0 开始。如果对给出的值没有指定键名，则取当前最大的整数索引值，而新的键名将是该值加一。如果指定的键名已经有了值，则该值会被覆盖。但自 PHP 4.3 起，若给一个当前最大键名是负值的数组添加一个新值，则新生成的索引为零(0)，而不再像以前的正值索引那样——新生成的索引为当前最大索引加一。

使用 TRUE 作为键名将使 integer 1 成为键名；使用 FALSE 作为键名将使 integer 0 成为键名；使用 NULL 作为键名将等同于使用空字符串；使用空字符串作为键名的话，将新建(或覆盖)一个用空字符串作为键名的值，这和用空的方括号不一样。

例 0410.php：

```
<?php
// 这个数组与下面的数组相同 ……
array(5 => 43, 32, 56, "b" => 12);

// ...
```

```
array(5 => 43, 6 => 32, 7 => 56, "b" => 12);
?>
```

例 0411.php：

```
<?php
// 一个包含两个元素的数组
$a[0] = "first";
$a[1] = "second";
$a[] = "third"; // 添加数组元素的简单方法
// 现在$a[2]被赋值为"third"
echo count($a),"\n"; // 打印出 3，因为该数组有 3 个元素
print_r($a);

// 用一条语句定义一个数组并赋值
$arr = array("sbabu" => "5348","keith" => "4829","carole" => "4533");
// 添加一个元素
$arr["dean"] = "5397";
// 如果你定义的 carale 元素错了，可更正
$arr["carole"] = "4522";
// 显示内容
print_r($arr);
//var_dump($arr);

echo "$arr[0]<br>\n"; // Notice: Undefined offset:  0
//重新索引
$arr = array_values($arr);
print_r($arr);
echo "$arr[0]<br>\n"; // 5348
echo "$arr[1]"; // 4829
?>
```

输出内容：

```
3
Array
(
    [0] => first
    [1] => second
    [2] => third
)
Array
(
    [sbabu] => 5348
    [keith] => 4829
```

```
    [carole] => 4522
    [dean] => 5397
)
<br>
Array
(
    [0] => 5348
    [1] => 4829
    [2] => 4522
    [3] => 5397
)
5348<br>
4829
```

其他对数组有用的函数如下：sort()、next()、prev()、each()。foreach 控制结构专用于数组，它提供了一个简单的方法来遍历数组，详见**“第 5 章 控制结构”**。

例 0412.php：

```
<?php
$arr = array(5 => 1, 12 => 2);
$arr[] = 56;    // 等同于 $arr[13] = 56;
$arr["x"] = 42; // 以 key "x"添加一个元素

print_r($arr);   // 4 个元素
unset($arr[5]); // 从数值 arr 中删除 key 为 5 的元素
print_r($arr);   // 3 个元素
unset($arr);    // 删除整个数组
print_r($arr);   // Notice: Undefined variable: arr
?>
```

输出内容：

```
Array
(
    [5] => 1
    [12] => 2
    [13] => 56
    [x] => 42
)
Array
(
    [12] => 2
    [13] => 56
    [x] => 42
)
```

❖ **注意：**

如上所述，如果给出方括号但没有指定键名，则取当前最大整数索引值，新的键名将是该值加一。如果当前还没有整数索引，则键名为 0。如果指定的键名已经有值了，该值将被覆盖。unset()函数允许删除数组中的某个键，但数组不会重建索引。

例 0413.php：

```
<?php
// 创建一个简单的数组
$arr = array(1, 2, 3, 4, 5);
print_r($arr);

// 现在删除其中的所有元素，但保持数组本身不变：
foreach ($arr as $i => $value) {
    unset($arr[$i]);
}
print_r($arr);

// 添加一个单元(注意新的键名是 5，而不是 0)
$arr[] = 6;
print_r($arr);

// 重新索引：
$arr = array_values($arr);
$arr[] = 7;
print_r($arr);
?>
```

输出内容：

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 4
    [4] => 5
)
Array
(
)
Array
(
    [5] => 6
```

```
)
Array
(
    [0] => 6
    [1] => 7
)
```

❖ 注意：

这里使用的最大整数键名，当前不一定就在数组中，它只要在上次数组重新生成索引后曾经存在过就行了。

6. 对象

使用 new 语句产生一个对象：

例 0414.php：

```
<?php
class foo
{
function do foo ()
{
echo "Doing foo.";
}
}
$bar = new foo;
$bar->do foo();      // 输出 Doing foo.
?>
```

如果将一个对象转换成对象，它将不会有任何变化。但如果其他任何类型的值被转换成对象，将会实例化一个内置类 stdClass 的对象。如果该值为 NULL，则新的实例为空。数组转换成对象，将使键名成为属性名并具有相对应的值。对于任何其他的值，名为 scalar 的成员变量将包含该值。

7. NULL

特殊的 NULL 值表示一个变量没有值。NULL 类型唯一可能的值就是 NULL。

在下列情况下，一个变量的值被认为是 NULL：被赋值为 NULL、尚未被赋值、被 unset()。

❖ 注意：

将一个变量转换为 NULL 类型，将会删除该变量并且 unset 它的值。可参考 is_null() 和 unset()。

8. 资源类型

资源是一种特殊变量，保存了到外部资源的一个引用。资源是通过专门的函数建立和使用的。资源类型在 PHP 4 版本时被引入，参见 get_resource_type()。

由于资源类型变量保存有为打开文件、数据库连接、图形画布区域等的特殊句柄，因此将其他类型的值转换为资源没有意义。

释放资源

由于 PHP 4 Zend 引擎引进了引用计数系统，可以自动检测到一个不再被引用了的资源(和 Java 一样)。这种情况下，此资源使用的所有外部资源都会被垃圾回收系统释放。因此，不再需要手工释放内存。但持久数据库连接比较特殊，它们不会被垃圾回收系统销毁。

9. 改变变量类型

PHP 手册中有一句话："PHP 不支持(也不需要)直接在声明变量时定义变量类型，变量类型将根据其被应用的情况决定。如果将变量 var 赋值为一个字符串，那么它将变成一个字符串。如果又为它赋予了整数值，那么它就变成了整数。"例如：

例 0415.php：

```
<?php
$foo = "0"; // $foo 是字符串(ASCII 48)
echo $foo."\n";
$foo++; // $foo 是字符串"1" (ASCII 49)
echo $foo."\n";
$foo += 1; // $foo 现在是整数(2)
echo $foo."\n";
$foo = $foo + 1.3; // $foo 是一个双精度数(3.3)
echo $foo."\n";
$foo = 5 + "10 Little Piggies"; // $foo 是一个整数(15)
echo $foo."\n";
$foo = 5 + "10 Small Pigs"; // $foo 是一个整数(15)
echo $foo."\n";
?>
```

输出内容：

```
0
1
2
3.3
15
15
```

如果想要强行转换变量类型，可以使用与 C 语言相同的函数 settype()。

4.3 常量与变量

1. 定义

常量是一个简单值的标识符(名称)。正如名称所示，在脚本执行期间该值不能改变(魔术常量除外，因为它们不是真正的常量)。常量默认为大小写敏感。通常常量总是大写。

变量用一个美元符号($)后接变量名称来表示。变量名称区分大小写。变量用于存储值，比如数字、文本字符串或数组。一旦设置了某个变量，我们就可以在脚本中重复使用它。

PHP 是一门松散类型的语言(Loosely Typed Language)，不需要在设置变量之前声明该变量。根据变量被设置的方式，PHP 会自动把变量转换为正确的数据类型。但在强类型的编程语言中，必须在使用前声明变量的类型和名称，如 Java、C++。

例 0416.php：

```
<?php
/* 判断常量是否存在*/
if (defined('MYCONSTANT')) {
echo MYCONSTANT;
}
//判断变量是否存在
if (isset($myvar)) {
echo "存在变量$myvar.";
}
//判断函数是否存在
if (function_exists('imap_open')) {
echo "存在函数 imag_open\n";
} else {
echo "函数 imag_open 不存在\n";
}
?>
```

2. 两者的相同之处

1) 名称以字母或底线开头，后接任意数目的字母、数字、下划线(也称底线)或多字节文字。用正则表达式可表示为：[a-zA-Z_\0x7f-\0xff][a-zA-Z0-9_\x7f-\xff]*(0x7f-0xff 为从 127 到 255 的 ASCII 字符)。任何 PHP 标签都遵循同样的命名规则。

2) 区分大小写，即对大小写敏感。

3. 两者的不同之处

1) 常量不像变量那样前面有美元符号($)，即常量前面没有美元符号($)。

2) 常量只能用 define() 函数定义，而不能通过赋值语句定义；在 PHP 5.3.0 以后，可以使用 const 关键字在类定义的外部定义常量。

3) 常量自被定义起其范围就是全局的，变量有特定的作用域，常量可以在任何地方定义和访问。

4) 常量一旦定义就不能被重新定义或取消定义，变量可重新赋值及取消定义。

5) 常量的值只能是标量(boolean、integer、float、string)，可以定义 resource 常量，但应尽量避免，因为会造成不可预料的结果。

6) 两者默认为大小写敏感，但按照惯例常量标识符总是大写。

7) 可以通过指定其名称来取得常量的值，如果常量名是动态的，可用函数 constant()来获取常量的值。用 get_defined_constants()可以获得所有已定义的常量的列表。

❖ **注意：**

常量和(全局)变量存放在不同的名称空间中。也就是说，TRUE 和$TRUE 是不同的。

4. 常量

`define("MYNAME","cnbruce")`表示定义了一个值为 cnbruce 的 MYNAME 常量。如果使用了一个未定义的常量，PHP 则假定想要的是该常量本身的名称，如同用字符串调用它一样(Constant 对应 "Constant")，此时还将发出一个 **E_NOTICE** 级的错误。

例 0417.php：

```
<?php
define("MYNAME","cnbruce");
$MYNAME="cnrose";
echo MYNAME."\n";    //输出"cnbruce"
echo $MYNAME."\n";   //输出"cnrose"
echo Constant;   //输出"Constant"，并发出一个 E NOTICE 提示性信息
?>
```

输出内容：

```
cnbruce
cnrose
Constant
```

例 0418.php：

```
<?php
// 以下代码在 PHP 5.3.0 之后版本中可以正常工作
```

```
const CONSTANT = "Hello World";
echo CONSTANT;  //输出" Hello World"
?>
```

如何将常量和变量的值一同输出？这需要用到 PHP 的字符串运算，使用英文句号(.)可将字符串连接合并成新的字符串，类似 ASP 中的&。

例 0419.php：

```
<?php
define("MYNAME","cnbruce");
$MYNAME="cnrose";

echo MYNAME.",".$MYNAME;     //输出为“cnbruce,cnrose”
?>
```

5. 魔术常量

PHP 向它运行的任何脚本均提供了大量的预定义常量，或称魔术常量，即不需要define()定义就能使用。但很多常量都是由不同的扩展库定义的，只有在加载了这些扩展库后才会出现，要么动态加载后，要么在编译时就已经包括进去了。

有 7 个魔术常量的值会随着它们在代码中位置的改变而改变，例如 __LINE__ 的值就依赖于它在脚本中所处的行。这些特殊的常量不区分大小写，如表 4-1 所示。

表 4-1　PHP 的几个“魔术常量”

名　称	说　明
__LINE__	文件中当前的行号
__FILE__	文件的完整路径和文件名。如果用在被包含文件中，则返回被包含文件的名称。自 PHP 4.0.2 起，__FILE__ 总是包含一个绝对路径(如果是符号连接，则是解析后的绝对路径)，而在此之前的版本中有时会包含一个相对路径
__DIR__	文件所在的目录。如果用在被包含文件中，则返回被包含文件所在的目录。它等价于 dirname(__FILE__)。除非是根目录，否则目录名不包括末尾的斜杠(PHP 5.3.0 新增)
__FUNCTION__	函数名称(PHP 4.3.0 新增)。自 PHP 5 起，该常量返回该函数被定义时的名称(区分大小写)。在 PHP 4 中，该值总是小写字母的
__CLASS__	类的名称(PHP 4.3.0 新增)。自 PHP 5 起，本常量返回该类被定义时的名称(区分大小写)。在 PHP 4 中，该值总是小写字母的
__METHOD__	类的方法名(PHP 5.0.0 新加)。返回该方法被定义时的名称(区分大小写)
__NAMESPACE__	当前命名空间的名称(大小写敏感)，这个常量是在编译时被定义的(PHP 5.3.0 新增)

__FILE__表示文件的完整路径和文件名，类似于 ASP 中的 Server.Mappath 当前文件。

例 0420.php：

```
<?php
echo   FILE  ;
?>
```

输出内容：

```
D:\howwe\wwwroot\0420.php
```

PHP 的预定义常量分为：

- 内核预定义常量，是指在 PHP 内核、Zend 和 SAPI 模块中定义的常量。
- 标准预定义常量，是指 PHP 中默认定义的常量。

6. 变量

虽然 PHP 中不需要初始化变量，但对变量进行初始化是个好习惯。未初始化的变量具有其所属类型的默认值：布尔类型的变量默认值为 FALSE，整型和符点型变量的默认值为零，字符串变量的默认值为空字符串，数组变量的默认值为空数组。

❖ **注意：**

$this 是一个特殊变量，它不能被赋值。

PHP 不支持(也不需要)在声明变量时直接定义变量的类型，变量的类型将根据其被应用的情况而定。变量都有一个美元符号($)作为前缀。所有变量都是局部变量，为了使在它所定义的函数中可以使用外部变量，可使用 global 语句。若要将该变量的作用范围限制在该函数之内，可使用 static 语句。

例 0421.php：

```
<?php
$g_var = 1 ; // 全局范围
function test()
{
global $g_var; // 这样就可以声明全局变量了
}
?>
```

变量默认为传值赋值，即将一个表达式的值赋予一个变量时，整个原始表达式的值都将被赋值到目标变量。

❖ **注意：**

当一个变量的值被赋予另一个变量时，改变其中一个变量的值，将不会影响到另一个变量。

PHP 3 变量总是传值赋值，PHP 4 提供了给变量赋值的另外一种方式——传地址赋值，也叫引用赋值。简单地追加一个符号(&)到将要赋值的变量前(源变量)，便意味着新的变量简单地引用了原始变量，改动新的变量将影响到原始变量，反之亦然。

例 0422.php 将输出“My name is Bob”两次：

```
<?php
$foo = 'Bob';             // 将 'Bob' 赋给 $foo
$bar = &$foo;             // 通过 $bar 引用 $foo
$bar = "My name is $bar";  // 修改 $bar 变量
echo $bar."\n";             // 输出 My name is Bob
echo $foo;                 // $foo 的值也将被修改，输出 My name is Bob
?>
```

输出内容：

```
My name is Bob
My name is Bob
```

变量 foo 只在首行被赋值，正常应输出“Bob”。然而在传址赋值给变量 bar 后，在变量 bar 的值发生变化的同时，变量 foo 的值也相应发生了变化。

请特别注意：只有有名称的变量才可以引用赋值。

例 0423.php：

```
<?php
$foo = 25;
$bar = &$foo;      // 合法的赋值
$bar = &(24 * 7);  // 非法；引用没有名称的表达式

function test()
{
   return 25;
}

$bar = &test();    // 非法
?>
```

变量的范围是它定义时的上下文环境(也就是它的生效范围)。大部分 PHP 变量只有一个单独的范围，但这个单独的范围跨度包含了 include 和 require 引入的文件，例如：

```
<?php
$a = 1;
include 'b.inc';
?>
```

这里，变量$a 将会在包含文件 b.inc 中生效。但是，在用户自定义函数中，一个局部

函数范围将被引入。任何用于函数内部的变量默认情况下都将被限制在局部函数范围内。

例 0424.php：

```
<?php
$a = 1; // 全局范围

function Test()
{
    echo $a; // 指向局部范围变量 Notice: Undefined variable: a
}

Test();
?>
```

上面这个脚本不会有任何输出，因为 echo 语句引用了一个局部版本的变量$a，而且在这个范围内，它并没有被赋值。可见 PHP 的全局变量和 C 语言不同，在 C 语言中，全局变量在函数中自动生效，除非被局部变量覆盖。但这在 C 语言中可能会引起一些问题，有些人可能不小心就改变了一个全局变量。为了避免这种情况发生，PHP 中全局变量在函数中使用时必须声明为 global，可参考例 0421.php。

7. (超)全局变量

PHP 全局变量是在引用变量时声明的，而非在程序首行定义，也不是在为变量赋值时才定义是全局变量还是局部变量。从 PHP 4.1.0 开始，PHP 提供了一套附加的预定数组，这些数组变量包含了来自 Web 服务器(假设可用)、运行环境和用户输入的数据。这些数组非常特别，它们在全局范围内自动生效，通常被称为自动全局(autoglobals)变量或超全局(super globals)变量。

❖ **注意：**

PHP 中没有用户自定义超全局变量的机制，旧的预定义数组($HTTP_*_VARS)仍旧存在。

例 0425.php：

```
<?php
$a = 1;
$b = 2;

function Sum()
{
global $a, $b;
$b = $a + $b;
}
```

```
Sum();
echo $b; // 输出 3
?>
```

输出内容：

```
3
```

如果函数 Sum()内没有使用 global 声明全局变量，程序便会对未定义的变量报错。$b 并不会被赋值，因为任何用于函数内部的变量默认情况下都将被限制在局部函数范围内。由于 Sum()函数中申明了全局变量$a 和$b，因而任何变量的所有引用变量都会指向全局变量。对于一个函数能够申明的全局变量的最大个数，PHP 没有限制。在全局范围内访问变量的第二种办法，是使用特殊的 PHP 自定义$GLOBALS 数组。前面的例子可以写成：

例 0426.php：

```
<?php
$a = 1;
$b = 2;

function Sum()
{
    $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
}

Sum();
echo $b;     // 输出 3
?>
```

$GLOBALS 是一个关联数组，每一个变量为一个元素，键名对应变量名，值对应变量的内容。$GLOBALS 之所以在全局范围内存在，是因为$GLOBALS 是一个超全局变量。

下面的例 0427.php 显示了超全局变量的用处：

```
<?php
function test_global()
{
    // 大多数的预定义变量并非超全局变量，它们需要用 global 关键字来使它们在函数的本地
       区域有效
    global $HTTP_POST_VARS;
    echo $HTTP_POST_VARS['name'];

    // Superglobals 在任何范围内都有效，它们并不需要 global 声明
    echo $_GET['name'];
}
test_global();
?>
```

在浏览器中打开页面 http://127.0.0.1/0427.php?name=howwe，则输出如下内容：

```
howwe
```

$_GET 为超全局变量，不需要用户定义，是 PHP 预定义的一些变量，其他的还有$_POST、$_COOKIE 等。

8. 静态变量

变量范围的另一个重要特性是静态变量(static variable)。静态变量仅在局部函数域中存在，但当程序执行离开此作用域时，其值并不丢失。看看下面的例子：

例 0428.php：

```
<?php
function Test()
{
    $a = 0;
    echo $a."\n";
    $a++;
}
Test();
Test();
?>
```

输出内容：

```
0
0
```

本函数没什么用，因为每次调用时都会将$a 的值设为 0 并输出"0"。将变量加一的$a++不起作用，因为一旦退出本函数变量$a 就不存在了。如果要写一个不会丢失本次计数值的计数函数，则需要将变量$a 定义为静态的。

例 0429.php(使用静态变量)：

```
<?php
function Test()
{
    static $a = 0;
    echo $a."\n";
    $a++;
}
Test();
Test();
?>
```

输出内容：

```
0
1
```

现在，变量$a 在第一次调用 test()时被初始化，每次调用 test() 函数都会输出 $a 的值并加一。

静态变量也提供了一种处理递归函数的方法。递归函数是一种调用自己的函数。写递归函数时要小心，因为可能会无穷递归下去。必须确保有充分的方法来终止递归。以下这个简单的函数递归计数到 10，使用静态变量$count 来判断何时停止。

例 0430.php(静态变量与递归函数)：

```
<?php
function Test()
{
    static $count = 0;

    $count++;
    echo $count." ";
    if ($count < 10) {
        Test();
    }
    $count--;
}
Test();
?>
```

输出内容：

```
1 2 3 4 5 6 7 8 9 10
```

静态变量可以按照上面例子中所示进行声明。如果在声明时用表达式的结果对其赋值，则会导致解析错误。

9. 可变变量

有时，使用可变变量名会很方便。也就是说，一个变量的名称可以动态地设置和使用。一个普通的变量通过声明来设置：

```
<?php
$a = 'hello';
?>
```

一个可变变量获取了一个普通变量的值，并将其作为这个可变变量的变量名。在上面的例子中，hello 使用了两个美元符号($)以后，就可以作为一个可变变量了：

```
<?php
$$a = 'world';
?>
```

这时，两个变量都被定义了：$a 的内容是“hello”，并且$hello 的内容是“world”。因此，可以表述为：

```
<?php
echo "$a ${$a}";
?>
```

以下写法更准确并且会输出同样的结果：

```
<?php
echo "$a $hello";
?>
```

它们都会输出“hello world”。

例 0431.php(将前面的内容综合起来，输出两个 hello world)：

```
<?php
$a = 'hello';
$$a = 'world';
echo "$a ${$a}"."\n";
echo "$a $hello";
?>
```

输出内容：

```
hello world
hello world
```

要将可变变量用于数组，就必须解决一个模棱两可的问题。就是写下$$a[1]时，解析器需要知道是想要$a[1]作为一个变量呢，还是想要$$a 作为一个变量并取出该变量中索引为[1]的值？解决此问题的方法：对第一种情况用${$a[1]}，对第二种情况用${$a}[1]。

❖ **注意：**

在 PHP 函数和类的方法中，超全局变量不能用作可变变量。

4.4 表达式

表达式是 PHP 最重要的内容。在 PHP 中，几乎所写的任何东西都是一个表达式。定

义一个表达式的方式是“任何有值的东西”，如最基本的表达式形式是常量和变量。

当键入“$a = 5”时，即表示将“5”分配给变量$a。“5”的值为 5，换句话说，“5”是一个值为 5 的表达式(在这里，“5”是一个整型常量)。赋值之后，所期待情况是$a 的值为 5，如写下$b = $a，则等同于$b = 5。换句话说，$b 是一个值也为 5 的表达式。

稍微复杂的表达式例子是函数。

例 0432.php：

```
<?php
function foo ()
{
    return 5;
}
?>
```

如果不熟悉函数的概念，请自己去 Google 或查看后续内容。键入$c = foo()，从本质上来说就如同写下$c = 5。

函数也是表达式，表达式的值即为函数的返回值。既然 foo()返回 5，那么表达式“foo()”的值也是 5。通常函数不会只返回一个静态值，而是进行一些计算。

PHP 支持 4 种标量值(标量值不能拆分为更小的单元，这和数组不同)：整型值 integer、浮点数值 float、字符串值 string 和布尔值 boolean。PHP 还支持两种复合类型：数组和对象。这两种类型都可赋值给变量或者从函数返回。

PHP 是一种面向表达式的语言，简单来说几乎一切都是表达式。前面的例子“$a = 5”涉及两个值，整型常量 5 的值以及变量$a 的值。并且$a 也被更新为 5，这意味着“$a = 5”是一个值为 5 的表达式。因而“$b = ($a = 5)”和“$a =5; $b=5”(分号标志着语句的结束)的效果是一样的。因为赋值操作的顺序是由右到左的，所以也可以写成“$b = $a =5”。

值得注意的表达式例子为前、后递增(++)和递减(--)。本质上来讲，前递增和后递增均增加了变量的值，并且对变量的影响是相同的，不同的是递增表达式的值。前递增写作“++$variable”，求增加后的值(PHP 在读取变量的值之前，增加变量的值，因而称为“前递增”)。后递增写作“$variable++”，求变量未递增之前的原始值(PHP 在读取变量的值之后，增加变量的值，因而称为“后递增”)。

一个常用到的表达式类型是比较表达式，这些表达式的值为 FALSE 或 TRUE。PHP 支持>(大于)、>=(大于等于)、==(等于)、!=(不等于)、<(小于)、<= (小于等于)等比较运算符。PHP 还支持全等运算符“===”(值和类型均相同)和非全等运算符“!==”(值或类型不同)。这些表达式都是条件判断语句(比如 if 语句)中最常用的。

有时，为了简化表达，使用组合运算赋值表达式。变量$a 加 1，可以简化为“$a++”或“++$a”。变量$a 增加大于 1(如 2)的值怎么简化呢？通用的做法是“$a = $a + 2”，最简单的方式是“$a += 3”，意思是“取变量$a 的值，加 3，得到的结果再次分配给变量$a”。这样做除了更简略和清楚外，还可更快地运行。

较为特别的表达式为三元条件运算符：$first ? $second : $third。

说明：如果第一个子表达式的值是 TRUE(非零)，那么计算第二个子表达式的值，其值即为整个表达式的值；否则，就将第三个子表达式的值作为整个表达式的值。

例 0433.php 有助于理解前、后递增和表达式：

```
<?php
function double($i)
{
   return $i*2;
}
$b = $a = 5;        // 将变量$a 和$b 赋值为 5
echo '$a:'.$a.' $b:'.$b."\n";

$c = $a++;          // 后递增将$c 赋值为 5
$e = $d = ++$b;     // 前递增将$d 和$e 赋值为 6
echo '$a:'.$a.' $b:'.$b.' $c:'.$c.' $d:'.$d.' $e:'.$e."\n";

$f = double($d++);  // 后递增将$f 赋值为 12(6*2)
$g = double(++$e);  // 前递增将$g 赋值为 14(2*7)
$h = $g += 10;      // $g+10 变为 24，为组合运算；再赋值给$h，$h 也为 24
echo '$a:'.$a.' $b:'.$b.' $c:'.$c.' $d:'.$d.' $e:'.$e.' $f:'.$f.' $g:'.$g.
     '$h:'.$h;
```

输出内容：

```
$a:5 $b:5
$a:6 $b:6 $c:5 $d:6 $e:6
$a:6 $b:6 $c:5 $d:7 $e:7 $f:12 $g:24 $h:24
```

4.5 运 算 符

运算符可以通过给出的一个或多个值(也称表达式)来产生另一个值(因而整个结构成为一个表达式)。可以简单地认为函数或任何会返回一个值(如 print)的结构是运算符，而那些没有返回值的(例如 echo)结构是非运算符。

有三种类型的运算符：第一种是一元运算符，只运算一个值，如!(取反运算符)和++(加一运算符)；第二种是有限二元运算符，PHP 支持的大多数运算符都是这种类型；第三种是三元运算符——?:，三元运算符是根据第一个表达式的值来在另外两个表达式的值中选择一个作为整个表达式的值，把整个三元表达式放在括号里更恰当些。

❖ **注意：**

三元运算符是条语句，其求值只能是语句的结果。如果想通过引用返回一个变量，将是无效的。假如在函数有这样的语句 return $var == 42 ? $a : $b;，它将不起作用。

1. 运算符的优先级

运算符的优先级指定了两个表达式绑定得有多“紧密”，如表达式 1 + 5 * 3 的结果是 16，而不是 18，因为乘号(*)的优先级比加号(+)高。必要时可用括号来强制改变优先级：(1 + 5) * 3 的值为 18。如果运算符的优先级相同，则采用从左到右的左联顺序。

表 4-2 从高到低列出了运算符的优先级，“左联”简称为“左”，表示表达式从左向右求值，右联则相反。同一行中的运算符具有相同优先级，此时它们的结合方向决定着求值顺序。

表 4-2　运算符优先级：从高到低

结 合 方 向	运　算　符	附 加 信 息
非结合	clone new	clone 和 new
左	[	array()
非结合	++/--	递增 / 递减运算符
非结合	(int)、(float)、(string)、(array)、(object)、(bool)	指定类型
非结合	Instanceof	类型
右结合	!	逻辑操作符
左	*、/、%	算术运算符
左	+、-、 .	算术/字符串运算符
左	<<、>>	位运算符
非结合	<、<=、>、>=、<>	比较运算符
非结合	==、!=、===、!==	比较运算符
左	&	位运算符和引用
左	^	位运算符
左	|	位运算符
左	&&	逻辑运算符
左	||	逻辑运算符
左	? :	三元运算符
右	=、+=、-=、*=、/=、.=、%=、&=、|=、^=、<<=、>>=	赋值运算符
左	and	逻辑运算符

(续表)

结合方向	运　算　符	附加信息
左	xor	逻辑运算符
左	or	逻辑运算符
左	,	(逗号)

❖ **注意：**

1) 逗号操作符用来分割函数参数和其他列表项，这个操作符经常被附带使用。

2) 使用括号可以增强代码的可读性。

3) 尽管“=”比其他大多数运算符的优先级低，PHP 仍旧允许类似如下所示的表达式：if (!$a = foo())。foo()的返回值被赋给了$a。

echo 中可用逗号来连接字符串，并且这样比直接用点号连接要快，但两者有区别。

例 0434.php：

```
<?php
echo 'abc'.'def';      //用点号连接字符串
echo "\n",'abc','def';    //用逗号连接字符串
echo "\n",'1+5=' . 1+5;   //输出 6
echo "\n",'1+5=' . 5+1;  //输出 2
echo "\n",('1+5' . 5)+1; //输出 2

echo "\n".'1+5=' , 5+1;  //输出 1+5=6
echo "\n".'1+5=' , 1+5;  //输出 1+5=6

echo "\n",'5+1=' . 1+5;  //输出 10
echo "\n",'5+1=' . 5+1;  //输出 6
echo "\n",'1+5=' . 1+5;  //输出 6
echo "\n",'1+5=' . 5+1;  //输出 2
?>
```

输出内容：

```
abcdef
abcdef
6
2
2
1+5=6
1+5=6
10
6
6
```

```
2
```

说明：用逗号分隔开就相当于是 N 个参数，即把 echo 当函数用。这样一来，echo 会对每个参数先进行计算，然后再进行连接，最后输出。

2. 算术运算符(参见表 4-3)

表 4-3　算术运算符

示　例	名　称	结　果
-$a	取反	$a 的负值
$a + $b	加法	$a 和 $b 的和
$a - $b	减法	$a 和 $b 的差
$a * $b	乘法	$a 和 $b 的积
$a / $b	除法	$a 除以 $b 的商
$a % $b	取模	$a 除以 $b 的余数

❖ **注意：**

1) 除法运算符一般返回浮点数。只有下列情况除外：两个操作数都是整数(或由字符串转换成的整数)并且正好能整除，这时就返回一个整数。

2) 取模运算符的操作数在运算之前都会转换成整数(除去小数部分)，取模运算 *$a % $b* 在 *$a* 为负值时结果也是负值。

3. 字符串运算符

有两个字符串运算符：第一个是连接运算符(.)，它返回其左右参数连接后的字符串；第二个是连接赋值运算符(.=)，它将右边参数附加到左边的参数后。更多信息请参见赋值运算符。

例 0435.php：

```
<?php
$a = "Hello ";
$b = $a . "World!"; // $b 的内容为"Hello World!"

$a = "Hello ";
$a .= "World!";     // $a 的内容为"Hello World!"
echo '$a:'.$a."\n";
echo '$b:'.$b;
?>
```

输出内容：

```
$a:Hello World!
$b:Hello World!
```

4. 赋值运算符

基本的赋值运算符是“=”。一开始可能会以为它是“等于”，实际上它是把右边表达式的值赋给左边的运算数。赋值运算表达式的值也就是所赋的值。

也就是说，“”的值是 3。这样就有了一些小技巧。

例 0436.php：

```
<?php
$a = 3; // $a 为 3
$a = ($b = 4) + 5; // $a 现为 9，而$b 为 4
?>
```

除基本赋值运算符之外，还有适合于所有二元算术、数组集合和字符串运算符的“组合运算符”，这样便可以在一个表达式中使用它的值并把表达式的结果赋给它。

例 0437.php：

```
<?php
$a = 3;
$a += 5; //sets $a to 8，等同于“$a = $a + 5;”
$b = "Hello ";
$b .= "There!"; //sets $b to "Hello There!"，等同于“$b = $b . "There!";”
?>
```

❖ 注意：

赋值运算将原变量的值拷贝到新变量中(传值赋值)，所以改变其中的一个并不影响另一个。这也适合于在很密集的循环中拷贝一些值，例如大数组。也可以使用引用赋值，采用$var = &$othervar;这样的语法。引用赋值意味着两个变量都指向同一个数据，没有任何数据的拷贝。在 PHP 5 中，对象总是采用引用赋值，除非明确使用新的 clone 关键字。

5. 数组运算符(参见表 4-4)

表 4-4 数组运算符

示 例	名 称	结 果
$a + $b	联合	$a 和$b 的联合
$a == $b	相等	如果$a 和$b 具有相同的键/值对，则为 TRUE

(续表)

示 例	名 称	结 果

$a === $b	全等	如果$a 和$b 具有相同的键/值对并且顺序和类型都相同，则为 TRUE
$a != $b	不等	如果$a 不等于$b，则为 TRUE
$a <> $b	不等	如果$a 不等于$b，则为 TRUE
$a !== $b	不全等	如果$a 不全等于$b，则为 TRUE

❖ **注意：**

“+” 运算符把右边的数组元素(除去键值与左边的数组元素相同的那些元素)附加到左边的数组后面，但是重复的键值不会被覆盖。

例 0438.php：

```
<?php
$a = array("a" => "apple", "b" => "banana");
$b = array("a" => "pear", "b" => "strawberry", "c" => "cherry");

$c = $a + $b; // Union of $a and $b
echo "Union of \$a and \$b: \n";
var_dump($c);

$c = $b + $a; // Union of $b and $a
echo "Union of \$b and \$a: \n";
var_dump($c);
?>
```

输出内容：

```
Union of $a and $b:
array(3) {
  ["a"]=>
  string(5) "apple"
  ["b"]=>
  string(6) "banana"
  ["c"]=>
  string(6) "cherry"
}
Union of $b and $a:
array(3) {
  ["a"]=>
  string(4) "pear"
  ["b"]=>
  string(10) "strawberry"
```

```
  ["c"]=>
  string(6) "cherry"
}
```

数组中的单元如果具有相同的键名和值，则比较时相等。

例 0439.php：

```
<?php
$a = array("apple", "banana");
$b = array(1 => "banana", "0" => "apple");
var_dump($a);
var_dump($b);

var_dump($a == $b); // bool(true)
var_dump($a === $b); // bool(false)
?>
```

输出内容：

```
array(2) {
  [0]=>
  string(5) "apple"
  [1]=>
  string(6) "banana"
}
array(2) {
  [1]=>
  string(6) "banana"
  [0]=>
  string(5) "apple"
}
bool(true)
bool(false)
```

6. 比较运算符(参见表 4-5)

比较运算符允许对两个值进行比较。

表 4-5 比较运算符

示　例	名　称	结　果
$a == $b	等于	TRUE，前提是 $a 等于 $b
$a === $b	全等	TRUE，前提是 $a 等于 $b，并且它们的类型也相同

(续表)

示　　例	名　　称	结　　果
$a != $b	不等	TRUE，前提是 $a 不等于 $b
$a <> $b	不等	TRUE，前提是 $a 不等于 $b
$a !== $b	非全等	TRUE，前提是 $a 不等于 $b 或者它们的类型不同
$a < $b	小与	TRUE，前提是 $a 严格小于 $b
$a > $b	大于	TRUE，前提是 $a 严格大于 $b
$a <= $b	小于等于	TRUE，前提是 $a 小于或等于 $b
$a >= $b	大于等于	TRUE，前提是 $a 大于或等于 $b

若要比较整数和字符串，则字符串会被转换为整数。若要比较两个数字字符串，则将它们作为整数进行比较。此规则也适用于 switch 语句。

例 0440.php：

```
<?php
var dump(0 == "a"); // 0 == 0 -> true
var_dump("1" == "01"); // 1 == 1 -> true
var dump("1" == "1e0"); // 1 == 1 -> true
switch ("a") {
case 0:
    echo "0";
    break;
case "a": // 分支无效，因为"a"等同于 0
    echo "a";
    break;
}
?>
```

对于多种类型的比较，比较运算符根据表 4-6 进行比较(按顺序)。

表 4-6　比较多种类型

运算数 1 的类型	运算数 2 的类型	结　　果
null 或 string	string	将 NULL 转换为""，进行数字或词汇比较
bool 或 null	任何其他类型	转换为 bool，FALSE<TRUE
object	object	内置类可以定义自己的比较，不同类不能比较，相同类和数组采用同样方式比较属性
String、number 或 resource	string、number 或 resource	将字符串和资源转换成数字，按普通数学比较

(续表)

运算数 1 的类型	运算数 2 的类型	结　果
array	array	具有较少成员的数组较小，如果运算数 1 中的键不存在于运算数 2 中，则数组无法比较，否则挨个值比较
array	任何其他类型	array 总是更大
object	任何其他类型	object 总是更大

例 0441.php(标准数组比较)：

```
<?php
// 数组是用标准比较运算符进行比较的
function standard_array_compare($op1, $op2)
{
    if (count($op1) < count($op2)) {
        return -1; // $op1 < $op2
    } elseif (count($op1) > count($op2)) {
        return 1; // $op1 > $op2
    }
    foreach ($op1 as $key => $val) {
        if (!array_key_exists($key, $op2)) {
            return null; // uncomparable
        } elseif ($val < $op2[$key]) {
            return -1;
        } elseif ($val > $op2[$key]) {
            return 1;
        }
    }
    return 0; // $op1 == $op2
}
?>
```

7. 位运算符(参见表 4-7)

位运算符允许对整型数中指定的位进行置位。如果左右参数都是字符串，则位运算符将操作字符的 ASCII 值。

表 4-7　位运算符

示　例	名　称	结　果
$a & $b	and(按位与)	将$a 和$b 中都为 1 的位设为 1
$a | $b	or(按位或)	将$a 或$b 中为 1 的位设为 1
$a ^ $b	xor(按位异或)	将$a 和$b 中不同的位设为 1

(续表)

示 例	名 称	结 果
~ $a	not(按位非)	将$a 中为 0 的位设为 1，为 1 的位设为 0
$a << $b	shift left(左移)	将$a 中的位向左移动$b 次(每一次移动都表示“乘以 2”)
$a >> $b	shift right(右移)	将$a 中的位向右移动$b 次(每一次移动都表示“除以 2”)

例 0442.php：

```
<?php
echo 12 ^ 9,"\n"; // 输出 '5'
echo "12" ^ "9","\n"; // 输出退格字符(ASCII 8)，('1'(ascii 49))^('9'(ascii
                         57))=#8
echo "hallo" ^ "hello","\n"; // 输出 ASCII 值 #0 #4 #0 #0 #0，'a' ^ 'e' = #4
echo 2 ^ "3","\n"; // 输出 1，2^((int)"3") == 1
echo "2" ^ 3,"\n"; // 输出 1，((int)"2") ^ 3 == 1
?>
```

8. 逻辑运算符(参见表 4-8)

表 4-8 逻辑运算符

示 例	名 称	结 果
$a and $b	and(逻辑与)	TRUE，如果 $a 与 $b 都为 TRUE 的话
$a or $b	or(逻辑或)	TRUE，如果 $a 或 $b 中任一为 TRUE 的话
$a xor $b	xor(逻辑异或)	TRUE，如果 $a 或 $b 中任一为 TRUE，但不同时是的话
! $a	not(逻辑非)	TRUE，如果 $a 不为 TRUE 的话
$a && $b	and(逻辑与)	TRUE，如果 $a 与 $b 都为 TRUE 的话
$a || $b	or(逻辑或)	TRUE，如果 $a 或 $b 任一为 TRUE 的话

例 0443.php：

```
<?php
// 下面的 foo() 不会被调用，因为它们被运算符“短路”了
$a = (false && foo());
$b = (true  || foo());
$c = (false and foo());
$d = (true  or  foo());

// "||" 的优先级比 "or" 高
$e = false || true; // $e 被赋值为 (false || true)，结果为 true
$f = false or true; // $f 被赋值为 false，因为"=" 的优先级比 "or" 高
var_dump($e, $f);
```

```
// "&&" 的优先级比 "and"  高
$g = true && false; // $g 被赋值为 (true && false)，结果为 false
$h = true and false; // $h 被赋值为 true，因为"=" 的优先级比 "and" 高
var_dump($g, $h);
?>
```

输出内容：

```
bool(true)
bool(false)
bool(false)
bool(true)
```

图 4-2 对常用的运算符作了总结。递增(++)/递减(--)运算符已在“4.4 表达式”中有所介绍，下面不再单独说明。错误控制运算符、执行运算符及类型运算符 instanceof 请自己去 Google，此处不作展开介绍。

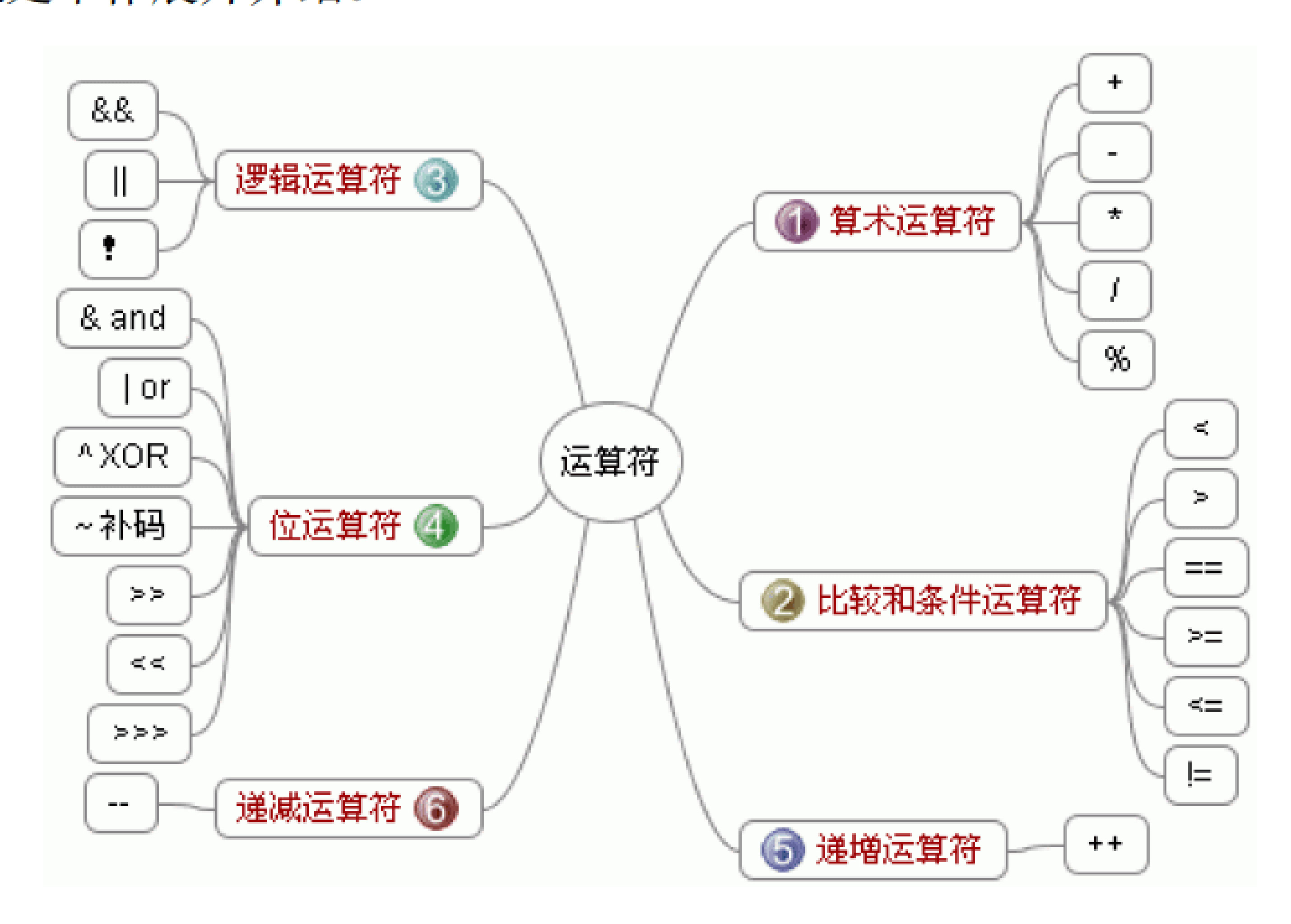


图 4-2　常用运算符

4.6　函　　数

可以像以下例子那样定义自己的函数，函数的返回值可以是任何数据类型。函数名和 PHP 中其他标识符的命名规则相同。

```
function foo (变量名一, 变量名二, . . . , 变量名 n)
{
echo "Example function.n";
```

```
return $retval;
}
```

所有 PHP 代码都可以出现在函数定义中，甚至包括对其他函数和类的定义。函数必须在引用之前定义。

PHP 中的所有函数和类都具有全局作用域，可以在内部定义外部调用，反之亦然。PHP 不支持函数重载，也不可能取消定义或者重定义已声明的函数。PHP 支持可变数量的参数和默认参数。PHP 可以调用递归函数，但要避免递归函数/方法调用超过 100 层，因为这样可能会破坏堆栈，导致当前脚本终止。

❖ **注意：**

函数名是大小写无关的，不过在调用函数的时候，通常使用与其定义时相同的形式。

1. 返回值

可通过可选的返回语句来返回值。可以返回包括数组和对象在内的任意类型。返回语句 return 会立即终止函数的运行，并将控制权交回调用该函数的代码行。函数不能返回多个值，但可以通过返回一个数组来得到与返回多个值类似的效果。从函数返回一个引用时，必须在函数声明和指派返回值给一个变量时都使用引用操作符“&”。

2. 可变函数

PHP 支持可变函数，这意味着如果一个变量名的后面跟有圆括号，PHP 将寻找与该变量的值同名的函数，并且尝试执行它。可变函数可以用来实现包括回调函数和函数表在内的一些用途。

可变函数不能用于语言结构，例如 echo()、print()、unset()、isset()、empty()、include()、require() 以及与此类似的语句。需要使用自己的包装函数，才能将这些结构用作可变函数。

例 0444.php：

```
<?php
function foo() {
    echo "In foo()<br />\n";
}

function bar($arg = '') {
    echo "In bar(); argument was '$arg'.<br />\n";
}

// 使用 echo 的包装函数
function echoit($string)
{
```

```
    echo $string;
}

$func = 'foo';
$func();        // This calls foo()

$func = 'bar';
$func('test');  // This calls bar()

$func = 'echoit';
$func('test');  // This calls echoit()
?>
```

输出内容：

```
In foo()<br />
In bar(); argument was 'test'.<br />
test
```

还可以利用可变函数的特性来调用一个对象的方法。

例 0445.php：

```
<?php
class Foo
{
    function Variable()
    {
        $name = 'Bar';
        $this->$name(); // This calls the Bar() method
    }

    function Bar()
    {
        echo "This is Bar";
    }
}

$foo = new Foo();
$funcname = "Variable";
$foo->$funcname();   // This calls $foo->Variable()
?>
```

输出内容：

```
This is Bar
```

❖ 注意：

匿名函数此处不作展开介绍，请自己去 Google。

4.7　计算机基础：原码、反码、补码

原码是机器数的一种简单的表示法。数值在计算机中的表示形式为机器数，计算机只能识别 0 和 1，使用二进制，但日常生活中使用的却是十进制。数值有正负之分，计算机用一个数的最高位存放符号(0 为正，1 为负)。

假设机器能处理的位数为 8，即字长为 1 个字节，则原码能表示的数值的范围为－127~－0 +0~127，共 256 个。

有了数值的表示方法后，就可以对数进行算术运算。当两数相加时，如果同号，则数值相加；如果异号，则进行相减。并且在进行减法时需要比较绝对值的大小，然后大数减去小数，最后还要为结果选择符号。直接运用这种方式去处理，计算机会很麻烦。

为了解决这些矛盾，人们找到了补码表示法，即数值一律用补码来表示(存储)。机器数的补码可由原码得到。如果机器数是正数，则该机器数的补码与原码一样；如果机器数是负数，则该机器数的补码是对它的原码各位(除符号位外)取反，并在末位加 1。

在补码中，用－128 代替了－0，所以补码的表示范围为－128~0~127，共 256 个。

❖ 注意：

－128 没有对应的原码和反码，(-128)＝(10000000)。

对除符号位之外的其余各位逐位取反，就得到了反码(对于正数，其反码与原码相同)。反码的取值空间和原码相同且一一对应。

负数用补码表示时，可以把减法转换为加法。这样，在计算机中实现起来就很方便。

补码的原理可以用钟表来描述：假设标准时间为 4 点整，一只表已经 7 点了，为了校准时间，可以采用两种方法 ——一是将时针退 7－4=3 格；一是将时针向前拨 12－3=9 格。即 7－3 和 7+9(mod12)等价。因此，把负数用补码表示的 mod2 操作，可以把减法转换为加法。使用补码，可以将符号位和其他位统一处理；同时，减法也可按加法来处理。另外，两个用补码表示的数相加时，如果最高位(符号位)有进位，则进位被舍弃。所以补码的设计目的是：

1) 使符号位能与有效值部分一起参与运算，从而简化运算规则。

2) 使减法运算转换为加法运算，进一步简化计算机中运算器的线路设计。

二进制补码的定义形式非常漂亮，因为它具有对称性。在二进制中，很多的东西都是“恰好如此”。恰好二进制只有 0、1 两个数，恰好只有正负两种状态……在补码的对应规则下，n 个数位能表示的数的个数还是那么多，只不过把其中一半的正数的形式让给了

负数。

```
正数:补码、反码、原码相同。
负数:补码是正数取反后加一。
最高位为符号位:0 代表正，1 代表负。

1 的 16 位编码 0000000000000001
-1 的编码为 1 的编码取反后+1:
1 的编码取反为:            1111111111111110
然后再加 1 为:             1111111111111111
这就是-1 的编码。

32767 的编码为:     0111111111111111
取反为:             1000000000000000
再加 1 得到-32767 的编码为: 1000000000000001
－32768 比 32767 还少 1，计算如下:
  1000000000000001
 -0000000000000001
------------------------
= 1000000000000000
```

第 5 章

控制结构

根据结构定理(structure theorem)，任何计算机程序都可以用图 5-1 中的基本控制结构来表示。可以对它们进行任意组合以解决问题。

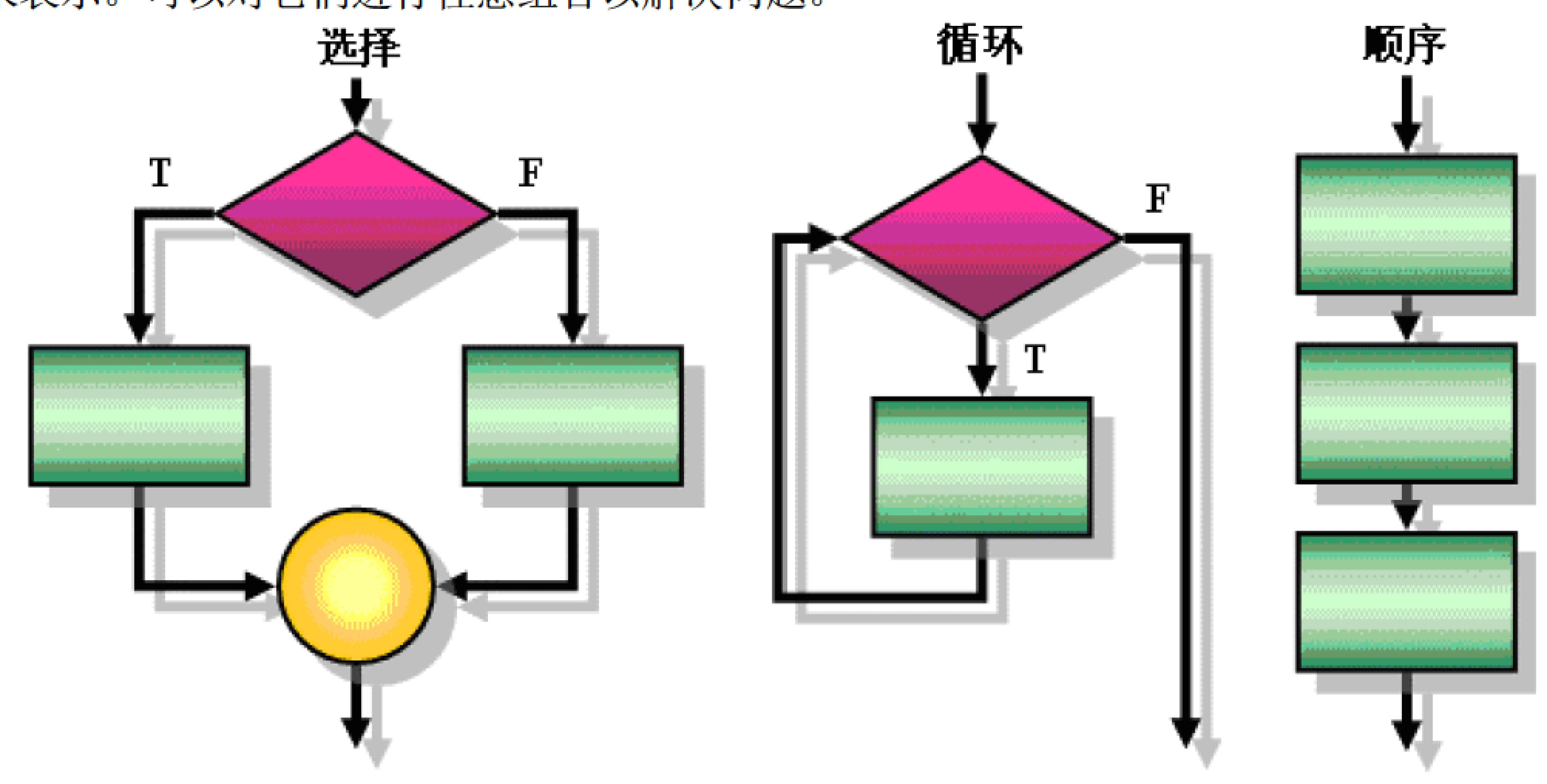


图 5-1 基本控制结构

- 选择结构(selection structrue)，用于测试条件，根据条件的真假，执行一系列语句。一个条件语句可以是任何能够返回布尔值(TRUE 或 FALSE)的变量或表达式。
- 循环结构(repetition structrue)，能在条件满足的情况下反复执行。
- 顺序结构(sequence structrue)，只是简单地按照顺序执行语句。

如图 5-2 所示，程序流程图就是控制结构的组合，但目前已经很少使用。

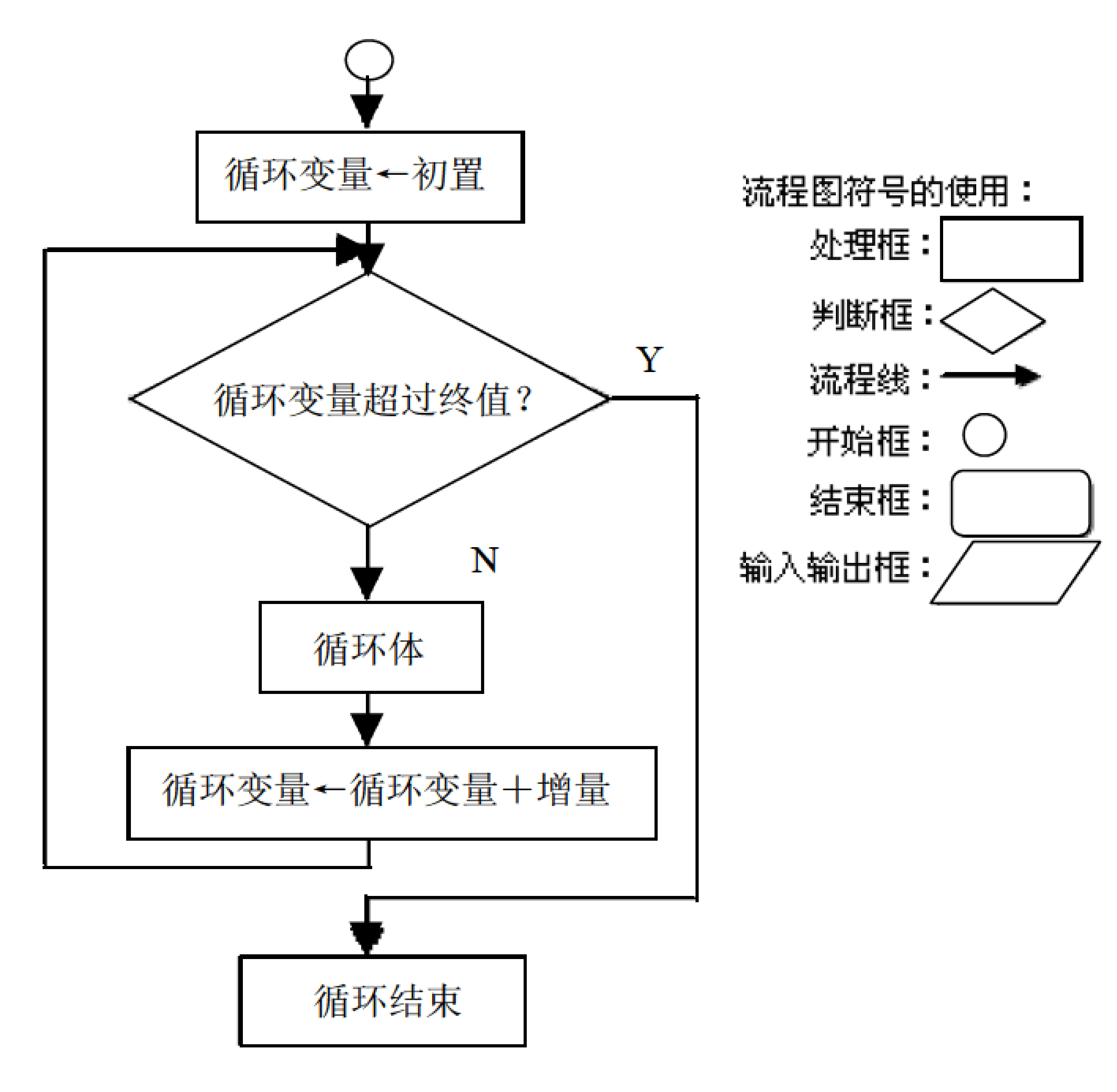


图 5-2　程序流程图

5.1　条 件 控 制

条件控制语句有两种：if 和 switch。

有时候，我们需要根据具体的条件来采取不同的对策。IF 语句就能让我们按条件来执行语句序列。也就是说，语句序列的执行与否取决于某个给定的条件。

有三种 IF 语句：if(单选)、if-else(多选)和 if-else if(多重选择)。

switch 语句是条件判断的精简形式，它能计算条件表达式的值并在多个对应策略中作出选择。

1. IF 结构

if 结构是很多语言(包括 PHP 在内)最重要的特性之一，它允许按照条件执行代码片段。PHP 的 if 结构和 C 语言相似：

```
<?php
if (expr)
    statement
?>
```

正如“**4.4 表达式**”一节所述，expr 的运算结果是布尔值。如果 expr 的值为 TRUE，

PHP 将执行 statement；如果值为 FALSE，将忽略 statement。

例 0501.php：

```
<?php
if ($a > $b) // 如果$a 大于$b，则显示 a is bigger than b
    echo "a is bigger than b";
?>
```

经常需要按照条件执行不止一条语句，可以将这些语句放入语句组中。几个语句组合在一起组成复合语句，在形式上是语句前后用一对花括号({})括起来，其中可以有一系列的定义或说明，后跟一系列的语句。为了养成良好的编码风格，建议即使只有一个语句，也要用一对花括号括起来。

例 0502.php：

```
<?php
if ($a > $b) {   // 如果$a 大于$b，则显示 a is bigger than b，并将$a 的值赋给$b
    echo "a is bigger than b";
    $b = $a;
}
?>
```

if 语句可以无限地嵌套在其他 if 语句中，这给程序的不同部分的条件执行提供了充分的弹性。else 的功能：if 语句只能用在满足某个条件时执行一条语句，要在不满足该条件时执行其他语句就必须用 else。else 延伸了 if 语句，可以在 if 语句中的表达式的值为 FALSE 时执行语句。

例 0503.php：

```
<?php
if ($a > $b) {   //当$a 大于$b 时，显示 a is bigger than b
    echo "a is bigger than b";
} else {//反之则显示 a is NOT bigger than b
    echo "a is NOT bigger than b";
}
?>
```

else 语句仅在 if 以及 elseif(如果有的话)语句中的表达式的值为 FALSE 时执行。

elseif(也可写成 else if)是 if 和 else 的组合。和 else 一样，它延伸了 if 语句，可以在原来的 if 表达式值为 FALSE 时执行不同语句。但和 else 不同的是，它仅在 elseif 的条件表达式值为 TRUE 时执行语句。

例 0504.php：

```
<?php
```

```
if ($a > $b) {  //当$a 大于$b 时，显示 a is bigger than b
    echo "a is bigger than b";
} elseif ($a == $b) {   //当$a 等于$b 时，显示 a equal to b
    echo "a is equal to b";
} else {//若上述都不满足，则显示 a is smaller than b
    echo "a is smaller than b";
}
?>
```

同一个 if 结构中可以有多个 elseif 语句。第一个表达式值为 TRUE 的 elseif 语句(如果有的话)将会执行，即 elseif 语句仅在之前的 if 或 elseif 的表达式值为 FALSE、而当前的 elseif 表达式值为 TRUE 时执行。

❖ **注意：**

elseif 与 else if 只有在类似上面使用大括号的情况下才认为是完全相同的。

例 0505.php：

```
<?php
/* Incorrect Method: */
if($a > $b):
    echo $a." is greater than ".$b;
else if($a == $b): // Will not compile.
    echo "The above line causes a parse error.";
endif;

/* Correct Method: */
if($a > $b):
    echo $a." is greater than ".$b;
elseif($a == $b): // Note the combination of the words.
    echo $a." equals ".$b;
else:
    echo $a." is neither greater than or equal to ".$b;
endif;
?>
```

流程控制的替代语法

PHP 提供了一些流程控制的替代语法，包括 if、while、for、foreach 和 switch。替代语法的基本形式是把左花括号({)换成冒号(:)，把右花括号(})换成 endif;、endwhile;、endfor;、endforeach;或 endswitch;。但一般不常用，故这里不作展开讨论。

2. switch 结构(多重选择)

switch 语句是控制选择的一种方式，编译器生成代码时将对这种结构进行特定的优化，从而产生效率比较高的代码，可读性高而且高效。如果可能的话，应尽量把 if-then-elseif 都改写成 switch 语句。

switch 语句和具有同样表达式的一系列的 if 语句相似，但 switch 会使用一个选择器，而不是多个布尔表达式。很多时候，需要把同一个变量(或表达式)与很多不同的值比较，并根据它等于哪个值来执行不同的代码，这就需要使用 switch 语句。

```
switch (表达式) {case 常量: 语句序列 case 常量: 语句序列 ...}
```

这里的“表达式”和“常量”可以是各种字符类型或整型。首先求“表达式”的值，然后用这个值与作为 switch 标号的各常量匹配。如果发现匹配的标号，就从这个位置开始执行。

switch 语句中的标号不能有相同的值。语句中可以有一个 default 标号，其他标号都不匹配时执行将转到 default 标号处。如果没有 default 标号，出现标号都不匹配的情况时语句将立即结束。人们通常在每个序列的最后放一个 break 语句。

❖ **注意：**

和其他语言不同，continue 语句放在 switch 结构中作用类似于 break。如果循环中有一个 switch 并希望 continue 到外层循环中的下一个轮回，可以使用 continue。

例 0506.php(使用 if 和 switch 语句实现相同功能)：

```
<?php
if ($i == 0) {
   echo "i equals 0";
} elseif ($i == 1) {
   echo "i equals 1";
} elseif ($i == 2) {
   echo "i equals 2";
}

switch ($i) {
   case 0:
      echo "i equals 0";
      break;
   case 1:
      echo "i equals 1";
      break;
   case 2:
      echo "i equals 2";
```

```
        break;
}
?>
```

例 0507.php(在 switch 结构中使用字符串)：

```
<?php
switch ($i) {
case "apple":
    echo "i is apple";
    break;
case "bar":
    echo "i is bar";
    break;
case "cake":
    echo "i is cake";
    break;
}
?>
```

switch 执行原理：switch 语句是一行接一行地执行(实际上是语句接语句地执行)，开始时没有代码被执行，仅当一个 case 语句中的值和 switch 表达式的值匹配时，PHP 才开始执行语句，直到 switch 的程序段结束或者遇到第一个 break 语句为止。如果不在 case 语句段最后写上 break，PHP 将继续执行下一个 case 中的语句段。

例 0508.php(无 break 语句的情况)：

```
<?php
switch ($i) {
    case 0:
        echo "i equals 0";
    case 1:
        echo "i equals 1";
    case 2:
        echo "i equals 2";
}
?>
```

如果$i 等于 0，PHP 将执行所有的 echo 语句！如果$i 等于 1，PHP 将执行后面两条 echo 语句。只有当$i 等于 2 时，才会得到“预期”的结果：只显示“i equals 2”。所以 break 语句很重要。

case 中的语句也可为空，这种情况下会将控制转移到下一个 case 中的语句。

例 0509.php(case 语句为空的情况)：

```
<?php
switch ($i) {
    case 0:
    case 1:
    case 2:
        echo "i is less than 3 but not negative";
        break;
    case 3:
        echo "i is 3";
}
?>
```

case 的特例是 default，它匹配了任何和其他 case 都不匹配的情况。

例 0509.php(case 为 default 语句的情况)：

```
<?php
switch ($i) {
    case 0:
        echo "i equals 0";
        break;
    case 1:
        echo "i equals 1";
        break;
    case 2:
        echo "i equals 2";
        break;
    default:
        echo "i is not equal to 0, 1 or 2";
}
?>
```

case 表达式可以是任何运算结果为简单类型的表达式，如整数、浮点数或字符串。不能是数组或对象，除非它们被解除引用成为简单类型。

❖ 提示：

有关 declare 结构的知识请自己去 Google。

5.2 循环控制

1. for 循环

for 循环是 PHP 中最复杂的循环结构，它的行为和 C 语言相似，用于执行重复性的指令。

for 循环的语法如下：

```
for (expr1; expr2; expr3)
    statement
```

expr1 为初始式，在循环开始前无条件求值一次。

expr2 为判断式，在每次循环开始前求值。值为 TRUE，则继续循环，执行嵌套的循环语句；值为 FALSE 则终止循环。

expr3 为递增式，在每次循环之后被求值(执行)。

如果 statement 语句只有一条，也就是非复合语句，那么花括号({ })可以省略不写。for 循环的第一条初始语句只会执行一次，之后每次重新进行循环时，都会根据判断式来判断是否执行下一个循环，而每次执行完循环之后，都会执行递增式。

每个表达式都可以为空或包括由逗号分隔的多个表达式。表达式 expr2 中，所有用逗号分隔的表达式都会计算，但只取最后一个结果。expr2 为空意味着将无限循环下去(和 C 一样，PHP 认为其值为 TRUE)，但必须用 break 语句来结束循环，而不是使用 for 的表达式真值判断。

例 0510.php(for 语句的 4 种方式，均显示数字 1 到 10)：

```
<?php
for ($i = 1; $i <= 10; $i++) {   // 方式 1
    echo $i;
}

for ($i = 1; ; $i++) {   // 方式 2
    if ($i > 10) {
        break;
    }
    echo $i;
}

$i = 1;
for (;;) {   // 方式 3
    if ($i > 10) {
        break;
    }
    echo $i;
    $i++;
}

for ($i = 1; $i <= 10; print $i, $i++);  // 方式 4
?>
```

方式 1(或方式 4)看上去最常见，但在 for 循环中用空表达式在很多场合下会很方便。

for 循环中也可以嵌套使用 for 循环。

例 0511.php(输出九九乘法表)：

```
<?php   //九九乘法表
for ($i=1;$i<=9;$i++)  //第一个循环,声明变量 i,循环 9 次，每次循环后 i 加 1
{
for ($j=1;$j<=$i;$j++){
    echo "$i*$j=".$i*$j." ";  //第二个循环，声明变量 j，循环次数将取决于 i 的值
}
echo "<br>\n";  // <br>用于换行，\n 则是空格
}

//两重循环合并，PHP 实现起来有些麻烦，可参考一下 Java 版的实现
/*
public class D10504NineTable {
    public static void main(String[] args) {
        int i,j;
        for (i=1;i<10;i++){
            for (j=1;j<=i;j++){
                System.out.printf("%d*%d=%2d%c", j, i, i * j, (i == j ? '\n' :
                                      ' '));
            }
        }

        //两重循环合并
        for (i = 1, j = 1; i<10&&j<=i; j=(j++==i)?((i++/i)+1):j) {
            System.out.printf("%d*%d=%2d%c", j, i, i * j, (i == j ? '\n' :
                                  ' '));
        }
    }
}
*/
?>
```

这个程序的目的在于让你看看 for 循环的写法。但原则上不鼓励这么写，程序要以清晰易懂为原则。

我们经常需要对下面这样的数组进行遍历：

例 0512.php(数组遍历)：

```
<?php
// 我们想要在遍历的过程中改变以下数组中某些元素的值
$people = Array(
```

```
        Array('name' => 'Kalle', 'salt' => 856412),
        Array('name' => 'Pierre', 'salt' => 215863)
        );
var dump($people);

for($i = 0; $i < sizeof($people); ++$i)
{
    $people[$i]['salt'] = rand(000000, 999999);
}
var_dump($people);
?>
```

输出内容：

```
array(2) {
  [0]=>
  array(2) {
    ["name"]=>
    string(5) "Kalle"
    ["salt"]=>
    int(856412)
  }
  [1]=>
  array(2) {
    ["name"]=>
    string(6) "Pierre"
    ["salt"]=>
    int(215863)
  }
}
array(2) {
  [0]=>
  array(2) {
    ["name"]=>
    string(5) "Kalle"
    ["salt"]=>
    int(524566)
  }
  [1]=>
  array(2) {
    ["name"]=>
    string(6) "Pierre"
    ["salt"]=>
    int(101379)
  }
}
```

上述粗体数字为随机内容。

但以上代码运行慢，因为 for 的第二个表达式会导致代码执行很慢，每次循环时都要计算一遍数组的长度。由于数组的长度始终不变，我们可以用一个中间变量来存储数组长度，然后将这个变量作为 for 循环的第二个表达式。这样在循环的时候就可以直接使用该变量的值，而不用每次都重新计算。

例 0513.php：

```
<?php
$people = Array(
        Array('name' => 'Kalle', 'salt' => 856412),
        Array('name' => 'Pierre', 'salt' => 215863)
        );
var_dump($people);

for($i = 0, $size = sizeof($people); $i < $size; ++$i)
{
    $people[$i]['salt'] = rand(000000, 999999);
}
var_dump($people);
?>
```

2. while 循环

while 循环是 PHP 中最简单的循环类型，和 C 语言中的 while 一样，它根据指定的条件判断是否执行循环体。

while 语句的基本格式如下：

```
while (expr)
    statement
```

如果循环体只有一条语句，则 while 的花括号({ })可以省略不写。while 等同于没有起始语句和终止语句的 for 循环，主要用于执行重复性的动作，而停止的时机是未知的。只要 while 表达式的值为 TRUE，就重复执行嵌套中的循环语句。表达式的值在每次开始循环时检查，所以即使这个值在循环语句中改变了，语句也不会停止执行，直到本次循环结束。如果 while 表达式的值一开始就是 FALSE，则循环语句一次都不会执行。

while 可以用作无穷循环，很多地方都要用到无穷循环。例如游戏设计中对使用者输入设备的轮询(Poll)，或是动画程序的播放等，都会使用到无穷循环。一个无穷循环如下所示：

```
while(true) {
    循环内容;
    ...
 }
```

无穷循环可以由自己循环中的某个条件式来结束。下面是一个循环内部终止的例子：

```
while(true) {
    语句;
    if(条件式)
        break;  // 跳离循环
     ...
}
```

当条件式成立时，会执行 break 离开 while 循环，这个 break 与在 switch 中的作用是一样的，都是在要离开当时执行的程序块时使用。

while 循环有时称为“当型循环”，因为它在循环执行前就会进行条件判断，而另一种 do while 循环称为“直到型循环”，它会先执行循环体，然后再进行条件判断。do while 与 while 的区别在于：前者表达式的值是在每次循环结束而不是开始时检查，这样能确保循环语句至少执行一次。

do while 的语法如下所示：

```
do {
    语句一;
    语句二;
    ...
 } while(条件式);
```

❖ **注意：**

while 后面是以分号(;)作为结束标志，这个经常被忽略。由于 do while 会先执行循环，所以它通常用于进行一些诸如初始化或接口通信这样的动作。

do while 扩展：把语句放在 do while(1)中，在循环内部用 break 语句结束循环的执行。

例 0514.php：

```
<?php
$i = 1;
while ($i <= 10) {   // 输出 1-10
    echo $i++;  // 后递增
}
echo "\n";
$i = 0;
```

```
do {// 循环只运行一次，第一次循环后，检查表达式时，值为 FALSE，因而终止
  echo $i;
} while ($i > 0);

echo "\n";
$i = 1;
do {// 输出 1-10
    if ($i > 10) {
        break;
    }
    echo $i++;  // 后递增
} while(1);
?>
```

输出内容：

```
12345678910
0
12345678910
```

3. foreach 循环

foreach 是一种遍历数组的简便方法，但仅能用于数组，当试图将其用于其他数据类型或一个未初始化的变量时会产生错误。

foreach 有两种语法，第二种相对次要但却是第一种的有力扩展。

```
foreach (array_expression as $value)
    statement
foreach (array_expression as $key => $value)
    statement
```

第一种格式将遍历给定的 array_expression 数组。每次循环时，当前单元的值被赋给 $value 并且数组内部的指针向前移一步(因此下一次循环时将会得到下一个单元)。

第二种格式的特别之处是还会将当前单元的键名在循环时赋给变量$key。

❖ **注意：**

1) 自 PHP 5 起，还可遍历对象。

2) 当 foreach 开始执行时，数组内部的指针会自动指向第一个单元，这意味着不需要在 foreach 循环之前调用 reset()。

3) 除非数组是被引用，否则 foreach 操作的是所指定数组的一个拷贝，而不是该数组本身。但对于数组指针来讲，除非对其重置，否则在 foreach 循环中或循环后都不要依赖数组指针的值。

4) 自 PHP 5 起，可以在$value 之前加上&来修改数组的元素。此方法将以引用来赋值而不是拷贝一个值。

5) foreach 不支持用“@”来抑制错误信息。

例 0515.php：

```
<?php   //此方法仅在被遍历的数组可以被引用时才可用
$arr = array(1, 2, 3, 4);
var dump($arr);
foreach ($arr as &$value) {
    $value = $value * 2;
}
var_dump($arr);
// $arr is now array(2, 4, 6, 8)
?>
```

例 0516.php(foreach 的替代用法)：

```
<?php
$arr = array("one", "two", "three");
reset($arr);
while (list(, $value) = each($arr)) {
    echo "Value: $value<br>\n";
}

foreach ($arr as $value) {
    echo "Value: $value<br />\n";
}

reset($arr);
while (list($key, $value) = each($arr)) {
    echo "Key: $key; Value: $value<br />\n";
}

foreach ($arr as $key => $value) {
    echo "Key: $key; Value: $value<br />\n";
}
?>
```

输出内容：

```
Value: one<br>
Value: two<br>
Value: three<br>
```

```
Value: one<br />
Value: two<br />
Value: three<br />
Key: 0; Value: one<br />
Key: 1; Value: two<br />
Key: 2; Value: three<br />
Key: 0; Value: one<br />
Key: 1; Value: two<br />
Key: 2; Value: three<br />
```

例 0517.php(更多的 foreach 范例)：

```
<?php
/* foreach example: value only */
$a = array(1, 2, 3, 17);
foreach ($a as $v) {
   echo "Current value of \$a: $v.\n";
}

/* foreach example: value (with its manual access notation printed for
    illustration) */
$a = array(1, 2, 3, 17);
$i = 0; /* for illustrative purposes only */
foreach ($a as $v) {
    echo "\$a[$i] => $v.\n";
    $i++;
}

/* foreach example: key and value */
$a = array(
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);
foreach ($a as $k => $v) {
    echo "\$a[$k] => $v.\n";
}

/* foreach example: multi-dimensional arrays */
$a = array();
$a[0][0] = "a";
$a[0][1] = "b";
$a[1][0] = "y";
```

```
$a[1][1] = "z";
foreach ($a as $v1) {
    foreach ($v1 as $v2) {
        echo "$v2\n";
    }
}

/* foreach example: dynamic arrays */
foreach (array(1, 2, 3, 4, 5) as $v) {
    echo "$v\n";
}
?>
```

输出内容：

```
Current value of $a: 1.
Current value of $a: 2.
Current value of $a: 3.
Current value of $a: 17.
$a[0] => 1.
$a[1] => 2.
$a[2] => 3.
$a[3] => 17.
$a[one] => 1.
$a[two] => 2.
$a[three] => 3.
$a[seventeen] => 17.
a
b
y
z
1
2
3
4
5
```

4. break 和 continue

break 可以离开当前 for、foreach、while、do while 的程序块以及 switch 结构，并前进至程序块后面的下一条语句，在 switch 中它主要用来中断与下一个 case 的比较。在 for、while、foreach 和 do while 中，它主要用来中断目前循环的执行。break 可以接受一个可选的数字参数以决定跳出几重循环。

continue 的作用与 break 类似，主要用于循环，所不同的是 break 会结束程序块的执行，而 continue 只会结束其之后程序块的语句，并跳回程序块的开头，在条件为真时继续下一个循环，而不是离开循环。continue 接受一个可选的数字参数，以决定跳过几重循环到循环结尾。

例 0518.php：

```
<?php
$arr = array('one', 'two', 'three', 'four', 'stop', 'five');
while (list (, $val) = each($arr)) {
    if ($val == 'stop') {
        break;    // 也可写成 'break 1;'
    }
    echo "$val<br />\n";
}

/* 使用可选参数 */
$i = 0;
while (++$i) {
    switch ($i) {
    case 5:
        echo "At 5<br />\n";
        break 1;  /* 只退出 switch. */
    case 10:
        echo "At 10; quitting<br />\n";
        break 2;  /* 退出 switch 和 while 循环 */
    default:
        break;
    }
}
?>
```

输出内容：

```
one<br />
two<br />
three<br />
four<br />
At 5<br />
At 10; quitting<br />
```

例 0519.php：

```
<?php
$arr = array('one', 'two', 'three', 'four', 'stop', 'five');
```

```
while (list ($key, $value) = each($arr)) {
    if (!($key % 2)) { // 跳过奇数情况
        continue;
    }
    echo '$key:'.$key.' $value:'.$value."<br />\n";
}

$i = 0;
while ($i++ < 5) {
    echo "Outer<br />\n";
    while (1) {
        echo "  Middle<br />\n";
        while (1) {
            echo "  Inner<br />\n";
            continue 3;
        }
        echo "This never gets output.<br />\n";
    }
    echo "Neither does this.<br />\n";
}
?>
```

输出内容：

```
$key:1 $value:two<br />
$key:3 $value:four<br />
$key:5 $value:five<br />
Outer<br />
  Middle<br />
  Inner<br />
Outer<br />
  Middle<br />
  Inner<br />
Outer<br />
  Middle<br />
  Inner<br />
Outer<br />
  Middle<br />
  Inner<br />
Outer<br />
  Middle<br />
  Inner<br />
```

5. return

return 返回语句有两种形式：

```
return;
return 表达式;
```

return 语句会立即结束当前函数的执行，控制返回函数调用处。第一种形式用于无返回值的函数，第二种形式用于有返回值的函数。第二种形式在执行时表达式被求值，得到的值转换成函数所要求的返回值类型，作为函数的返回值。

如果在一个函数中调用 return 语句，将立即结束此函数的执行并将它的参数作为函数的值返回。return 还能终止 eval 语句或脚本文件的执行。

如果是在全局范围中调用，则当前脚本文件终止运行。如果当前脚本文件是被 include()或 require()的，则控制交回调用文件。此外，如果当前脚本是被 include()的，则 return 的值会被当作 include()调用的返回值。如果在主脚本文件中调用 return，则脚本终止运行。如果当前脚本文件是由 php.ini 中的配置选项 auto_prepend_file 或 auto_append_file 指定的，则此脚本文件终止运行。

❖ **注意：**

return 是语言结构而不是函数，因此没有必要用括号。

5.3 包含文件

1. include()

include()语句包含并运行指定文件，但需要设置合适的 include_path。

以下内容也适用于 require()，两者除了如何处理失败之外其他完全一样。include()产生一个警告，而 require()则导致一个致命错误。也就是说，如果想在遇到丢失文件情况时停止处理页面，就用 require()，用 include()时脚本会继续运行。

寻找包含文件的顺序：先在当前工作目录相应的 include_path 下寻找，然后在当前运行脚本所在目录相应的 include_path 下寻找。例如 include_path 的当前工作目录是 /www/，脚本中要 include 一个 include/a.php，并且该文件中有一句 include "b.php"，则寻找 b.php 的顺序先是/www/，然后是/www/include/。如果文件名以“./”或“../”开始，则只在当前工作目录相应的 include_path 下寻找。

当一个文件被包含时，其中所包含的代码继承了 include 所在行的变量范围。从此处开始，调用文件在该行处可用的任何变量在被调用文件中都可用，不过所有在包含文件中

定义的函数和类都应具有全局作用域。

例 0520vars.php：

```
<?php
$color = 'green';
$fruit = 'apple';
?>
```

例 0521.php：

```
<?php
echo "A $color $fruit\n"; // 输出 A
include '0520vars.php';

echo "A $color $fruit"; // 输出 A green apple
?>
```

如果 include 出现在调用文件的一个函数中，则被调用文件中所包含的所有代码将表现得如同它们是在该函数内部定义的一样，因而它将遵循该函数的变量范围。

例 0522.php：

```
<?php
function foo()
{
    global $color;
    include '0520vars.php';
    echo "A $color $fruit";
}

/* 0520vars.php 在函数 foo()内，在函数外$fruit 将无效，$color 因声明为全局变量而有效*/
foo();                      // A green apple
echo "\nA $color $fruit";   // A green
?>
```

输出内容：

```
A green apple
A green
```

当一个文件被包含时，语法解析器在目标文件的开头脱离 PHP 模式并进入 HTML 模式，到文件结尾处恢复。因此，目标文件中应被当作 PHP 代码执行的任何代码，都必须被包括在有效的 PHP 起始和结束标记之中。

如果“URL fopen wrappers”在 PHP 中被激活(默认配置)，则可以用 URL(通过 HTTP 或其他支持的封装协议)而不是本地文件来指定要被包含的文件。如果目标服务器将目标文

件作为 PHP 代码解释，则可以用适用于 HTTP GET 的 URL 请求字符串来向被包括的文件传递变量。该脚本文件实际上已经在远程服务器上运行了，而本地脚本只包括其结果。远程文件的详细内容请自己去 Google。

处理返回值：可以在被包括的文件中使用 return 语句来终止该文件中程序的执行，并返回调用它的脚本，同样也可以从被包含的文件中返回值，还可以像普通函数一样获得 include 调用的返回值。但这在包含远程文件时可能不行，除非远程文件的输出具有合法的 PHP 开始和结束标记，就像任何本地文件一样。可以在标记内定义所需的变量，该变量在文件被包含的位置之后可用。

例 0523.php：

```
<?php
// won't work, evaluated as include(('0520vars.php') == 'OK'), i.e. include('')
if (include('0520vars.php') == 'OK') {
    echo 'OK';
}

// works
if ((include '0520vars.php') == 'OK') {
    echo 'OK';
}
?>
```

例 0524return.php：

```
<?php
$var = 'PHP';
return $var;
?>
```

例 0525noreturn.php：

```
<?php
$var = 'PHP';
?>
```

例 0526.php：

```
<?php
$foo = include '0524return.php';
echo $foo; // 输出'PHP'

$bar = include '0525noreturn.php';
echo "\n".$bar; // 输出 1
?>
```

$bar 的值为 1，是因为 include 已成功运行，返回 TRUE。如果文件不能被包含，则返回 FALSE 并发出一个 E_WARNING 警告。

如果包含文件中定义有函数，那么这些函数在 return()之前可在主文件中使用，否则不行。如果文件被包含两次，PHP 5 将发出致命错误警告，因为函数已经被定义。但 PHP 对 return 之后的函数不会处理，即不会检查是否已被定义。推荐使用 include_once()，这样便不会检查文件是否已包含并在包含文件之中，而是有条件地返回。

相关内容有：require()、require_once()、include_once()、get_included_files()、readfile()、virtual()和 include_path。

2. require()

require()语句包含并运行指定文件。更多例子请参考前面的 include()。

❖ 注意：

循环结构不影响 require()的行为，尽管目标文件中包含的代码仍然是循环的主体，但 require()本身只会运行一次。

3. require_once()

require_once()语句在脚本执行期间包含并运行指定文件，作用和 require()语句完全相同，唯一区别是如果该文件中的代码已经被包含，则不会再次包含。

require_once()语句用于在脚本执行期间，同一个文件有可能被包含超过一次的情况下，确保它只被包含一次以避免函数重定义、变量重新赋值等问题发生。

4. include_once()

include_once()语句在脚本执行期间包含并运行指定文件，作用和 include()语句类似，唯一区别是如果该文件中的代码已经被包含，则不会再次包含。

include_once()语句用于在脚本执行期间同一个文件有可能被包含超过一次的情况下，想确保它只被包含一次以避免函数重定义、变量重新赋值等问题发生。

返回值和 include()相同。如果文件已被包含，则 include_once()函数返回 TRUE。

5. goto

goto 操作符可以用来跳转到程序中的某一指定位置。该目标位置可以用目标名称加冒号来标记。PHP 中的 goto 有一定限制，只能在同一个文件和作用域中跳转，无法跳出一个函数或类方法，也无法跳入另一个函数，更不能跳入任何循环或 switch 结构中。常见的用法是跳出循环或 switch 结构，以代替多层的 break。

例 0527.php：

```
<?php
goto a;
echo 'Foo';

a:
echo 'Bar';
?>
```

输出内容：

```
Bar
```

例 0528.php：

```
<?php
for($i=0,$j=50; $i<100; $i++) {
  while($j--) {
    if($j==17) goto end;
  }
}
echo "i = $i";
end:
echo 'j hit 17';
?>
```

输出内容：

```
j hit 17
```

以下写法无效：

```
<?php
goto loop;
for($i=0,$j=50; $i<100; $i++) {
  while($j--) {
    loop:
  }
}
echo "$i = $i";
?>
```

❖ **注意：**

goto 操作符仅在 PHP 5.3 及以上版本中有效。

第 6 章

数 据 结 构

在计算机科学领域，数据结构(Data Structure)是指计算机中存储及组织数据的方式。数据结构是相互之间存在一种或多种特定关系的数据元素的集合。

通常情况下，精心选择的数据结构可以带来效率最优的算法。一般而言，数据结构的选择首先会从抽象数据类型的选择开始。一个设计良好的数据结构，应该在尽可能使用较少的时间与空间资源的前提下，为各种临界状态下的运行提供支持。数据结构可通过编程语言提供的数据类型、引用及其他操作加以实现。

不同种类的数据结构适合于不同种类的应用，而部分甚至专门用于特定的作业任务。例如，当计算机网络依赖路由表而运作时，B 树就可用于数据的封装。

在程序设计中，选择适当的数据结构是一个主要的考虑因素。许多大型系统的架构经验表明，封装的困难程度与最终成果的质量及表现，都取决于是否选择了最优的数据结构。在许多时候，确定了数据结构后便能很容易地得到算法。而有些时候，方向则会颠倒过来：例如当某个关键作业需要特定数据结构下的算法时，会反过来确定其所要使用的数据结构。然而，不管是哪种情况，数据结构的选择都是至关重要的。

系统架构的关键因素是数据结构而非算法，这导致了多种形式化的设计方法与编程语言的出现。绝大多数的语言都带有某种程度上的模块化思想，通过将数据结构的具体实现封装隐藏在接口里面的方法中，来让不同的应用程序能够安全地重用这些数据结构。C++、Java、Python 等面向对象的程序设计语言可使用类来完成这一功能。

因为数据结构的重要性毋庸置疑，现代编程语言及其运行环境在标准库中都包含了多种的数据结构，例如 C++标准模板库中的容器、Java 集合框架以及微软的.NET Framework。

大多数数据结构都由数列、记录、可辨识联合、引用等基本类型构成。举例而言，可空引用(nullable reference，一种可被置空的引用)是引用与可辨识联合的结合体，而最简单的链式结构——链表，则是由记录与可空引用构成。

数据结构意味着接口或封装：一个数据结构可被视为两个函数之间的接口，或者是

由多个数据类型联合组成的、用来存储内容的访问方法的封装。

PHP 参照了 C 语言的很多语法，不过数据结构对于一般的应用没多大影响，不用花费太多的精力。

6.1 基 本 概 念

Lobert L.Kruse 在《数据结构与程序设计》一书中，将一个数据结构的设计过程分成抽象层、数据结构层和实现层。其中，抽象层是指抽象数据类型层，讨论数据的逻辑结构及运算；数据结构层和实现层讨论数据结构的表示及其在计算机中的存储细节，以及运算的实现。

一个数据结构是由数据元素依据某种逻辑联系(或关系)组织起来的。对数据元素间逻辑关系的描述称为数据的逻辑结构；数据必须在计算机中存储，数据的存储结构是数据结构的实现形式，是其在计算机中的表示；此外，讨论一个数据结构必须同时讨论在该类数据上执行的运算，这样才有意义。

在许多类型的程序的设计中，数据结构的选择是一个基本的设计考虑因素。许多大型系统的架构经验表明，系统实现的困难程度和系统架构的质量都严重依赖于是否选择了最优的数据结构。许多时候，确定了数据结构后，算法就容易得到了。有些时候则相反，我们需要根据特定算法来选择数据结构与之适应。不论哪种情况，选择合适的数据结构都是非常重要的。

数据(Data)是信息的载体，它能够被计算机识别、存储和加工处理。它是计算机程序加工的原料，应用程序处理各种各样的数据。在计算机科学领域，数据就是计算机加工处理的对象，它可以是数值数据，也可以是非数值数据。数值数据是一些整数、实数或复数，主要用于工程计算、科学计算和商务处理等；非数值数据包括字符、文字、图形、图像、语音等。

数据元素(Data Element)是数据的基本单位。在不同的条件下，数据元素又可称为元素、结点、顶点、记录等。例如，学生信息检索系统中学生信息表中的一个记录、八皇后问题中状态树的一个状态、教学计划编排问题中的一个顶点等，都被称为一个数据元素。

有时，一个数据元素可由若干个数据项(Data Item)组成。例如，学籍管理系统中学生信息表的每一个数据元素就是一个学生记录，它包括学生的学号、姓名、性别、籍贯、出生年月、成绩等数据项。这些数据项可以分为两种：一种叫做初等项，如学生的性别、籍贯等，这些数据项是数据处理时不能再分割的最小单位；另一种叫做组合项，如学生的成绩，它可以再划分为数学、物理、化学等更小的项。通常，在解决实际应用问题时，把每个学生记录当作一个基本单位进行访问和处理。

数据的概念如图 6-1 所示，图 6-1 显示了数据与数据元素的关系。数据示例如图 6-2

所示，它显示了学生信息表的构成。

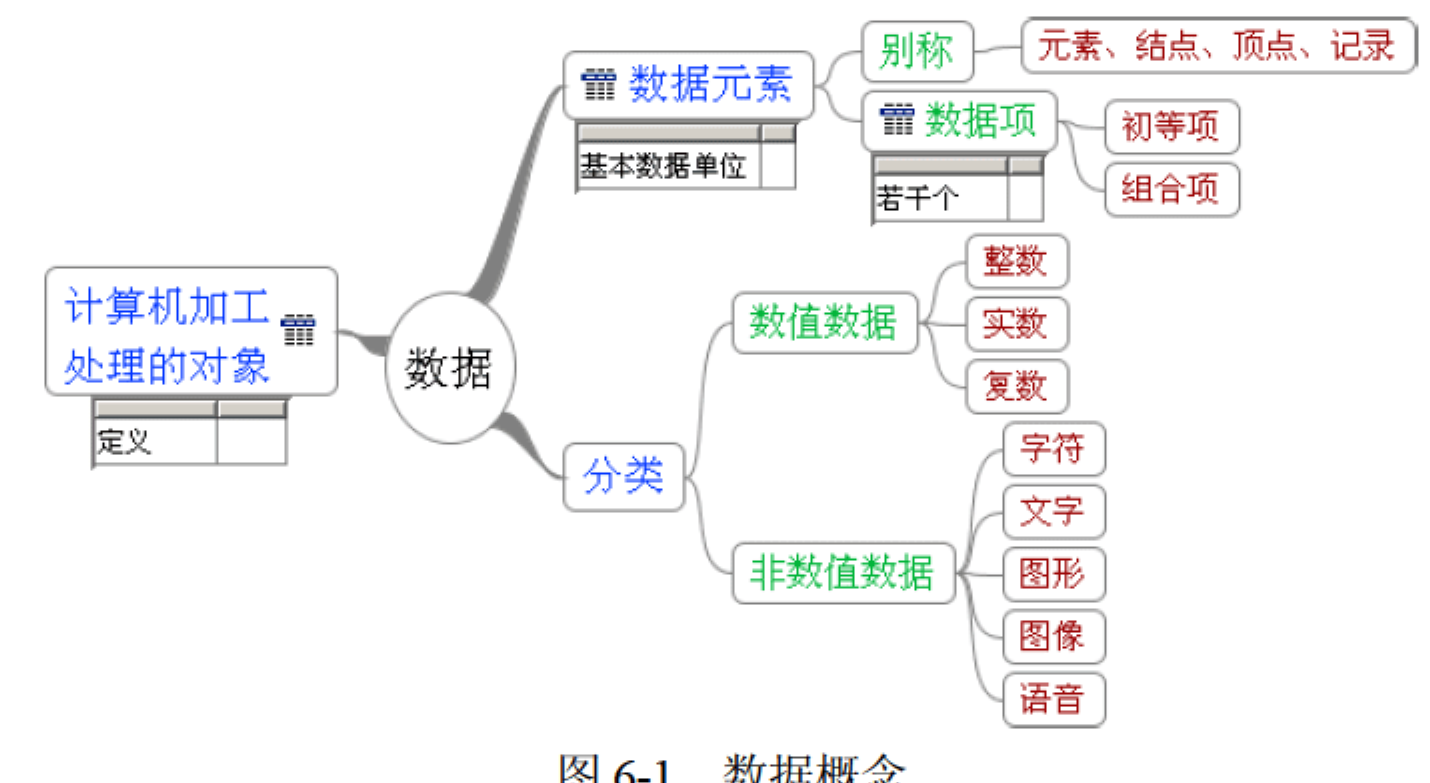


图 6-1 数据概念

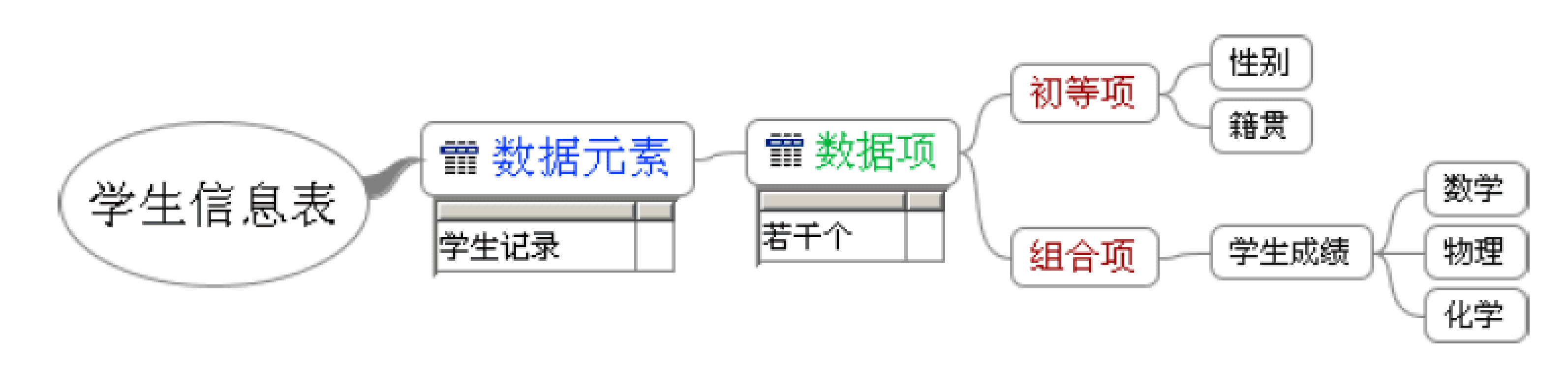


图 6-2 数据范例

数据对象(Data Object)或数据元素类(Data Element Class)是具有相同性质的数据元素的集合。在某个具体问题中，数据元素都具有相同的性质(元素值不一定相等)，属于同一数据对象(数据元素类)，数据元素是数据元素类的一个实例。例如，在交通咨询系统的交通网中，所有的顶点都属于一个数据元素类，顶点 A 和顶点 B 各自代表一个城市，是该数据元素类中的两个实例，其数据元素的值分别为 A 和 B。

数据结构(Data Structure)是指互相之间存在着一种或多种关系的数据元素的集合。在任何问题中，数据元素之间都不会是孤立的，它们之间都存在着这样或那样的关系，这种数据元素之间的关系称为结构。根据数据元素间关系的不同特性，通常有下列 4 类基本的结构：

1) 集合结构。在集合结构中，数据元素间的关系是“属于同一个集合”。集合是元素关系极为松散的一种结构。

2) 线性结构。该结构的数据元素之间存在着一对一的关系。

3) 树型结构。该结构的数据元素之间存在着一对多的关系。

4) 图形结构。该结构的数据元素之间存在着多对多的关系，图形结构也称作网状结构。

6.2 基础数据结构

几种基本的线性结构：线性表、栈、队列、串等。几种非线性结构：多维数组、广

义表、树、图、矩阵、哈希表、堆、优先队列、空间数据结构等。常用的数据处理技术：插入、删除、排序、查找等，文件的存储结构和组织方法等。

但在实际编程工作中，大部分数据结构都不会用到，而且也许永远都不会用到。尽管如此，深入地理解基本数据结构的概念和实现细节，仍然是每一个程序员的任务。这是因为，掌握这些知识将有利于更加正确和灵活地应用它们。

下面仅列出几种基本的数据结构算法，可仔细参考其实现思路，以及它们的使用技巧。

1. 二分查找(在数组中查找某个元素)

```
function bin_sch($array, $low, $high, $k){
    if ($low <= $high){
        $mid = intval(($low+$high)/2);
        if ($array[$mid] == $k){
            return $mid;
        }elseif ($k < $array[$mid]){
            return bin_sch($array, $low, $mid-1, $k);
        }else{
            return bin_sch($array, $mid+1, $high, $k);
        }
    }
    return -1;
}
```

2. 顺序查找(在数组中查找某个元素)

```
function seq_sch($array, $n, $k){
    $array[$n] = $k;
    for($i=0; $i<$n; $i++){
        if($array[$i]==$k){
            break;
        }
    }
    if ($i<$n){
        return $i;
    }else{
        return -1;
    }
}
```

3. 线性表的删除(在数组中实现)

```
function delete_array_element($array, $i)
{
        $len = count($array);
        for ($j=$i; $j<$len; $j++){
                $array[$j] = $array[$j+1];
        }
        array_pop($array);
        return $array;
}
```

4. 冒泡排序(数组排序)

```
function bubble sort($array)
{
        $count = count($array);
        if ($count <= 0) return false;

        for($i=0; $i<$count; $i++){
                for($j=$count-1; $j>$i; $j--){
                        if ($array[$j] < $array[$j-1]){
                                $tmp = $array[$j];
                                $array[$j] = $array[$j-1];
                                $array[$j-1] = $tmp;
                        }
                }
        }
        return $array;
}
```

5. 快速排序(数组排序)

```
function quick_sort($array) {
        if (count($array) <= 1) return $array;

        $key = $array[0];
        $left arr = array();
        $right arr = array();

        for ($i=1; $i<count($array); $i++){
                if ($array[$i] <= $key)
```

```
                $left_arr[] = $array[$i];
            else
                $right_arr[] = $array[$i];
        }

        $left_arr = quick_sort($left_arr);
        $right_arr = quick_sort($right_arr);

        return array_merge($left_arr, array($key), $right_arr);
}
```

6. 字符串处理

```
//字符串长度
function strlen($str)
{
        if ($str == '') return 0;

        $count = 0;
        while (1){
            if ($str[$count] != NULL){
                $count++;
                continue;
            }else{
                break;
            }
        }
        return $count;
}

//字符串翻转
function strrev($str)
{
        if ($str == '') return 0;
        for ($i=(strlen($str)-1); $i>=0; $i--){
            $rev str .= $str[$i];
        }
        return $rev str;
}

//字符串比较
function strcmp($s1, $s2)
{
```

```
        if (strlen($s1) < strlen($s2)) return -1;
        if (strlen($s1) > strlen($s2)) return 1;

        for ($i=0; $i<strlen($s1); $i++){
               if ($s1[$i] == $s2[$i]){
                      continue;
               }else{
                      return false;
               }
        }
        return 0;
}

//查找字符串
function strstr($str, $substr)
{
        $m = strlen($str);
        $n = strlen($substr);
        if ($m < $n) return false;

        for ($i=0; $i<=($m-$n+1); $i++){
               $sub = substr($str, $i, $n);
               if (strcmp($sub, $substr) == 0)  return $i;
        }
        return false;
}

//字符串替换
function str_replace($substr, $newsubstr, $str)
{
        $m = strlen($str);
        $n = strlen($substr);
        $x = strlen($newsubstr);
        if (strchr($str, $substr) == false) return false;

        for ($i=0; $i<=($m-$n+1); $i++){
               $i = strchr($str, $substr);
               $str = str_delete($str, $i, $n);
               $str = str_insert($str, $i, $newstr);
        }
        return $str;
}
```

```
//插入一段字符串
function str_insert($str, $i, $substr)
{
        for($j=0; $j<$i; $j++){
                $startstr .= $str[$j];
        }
        for ($j=$i; $j<strlen($str); $j++){
                $laststr .= $str[$j];
        }
        $str = ($startstr . $substr . $laststr);

        return $str;
}

//删除一段字符串
function str delete($str, $i, $j)
{
        for ($c=0; $c<$i; $c++){
                $startstr .= $str[$c];
        }
        for ($c=($i+$j); $c<strlen($str); $c++){
                $laststr .= $str[$c];
        }
        $str = ($startstr . $laststr);

        return $str;
}

//复制字符串
function strcpy($s1, $s2)
{
        if (strlen($s1)==NULL || !isset($s2)) return;

        for ($i=0; $i<strlen($s1); $i++){
                $s2[] = $s1[$i];
        }
        return $s2;
}

//连接字符串
function strcat($s1, $s2)
{
        if (!isset($s1) || !isset($s2)) return;
```

```
        $newstr = $s1;
        for($i=0; $i<count($s); $i++){
               $newstr .= $st[$i];
        }
        return $newsstr;
}
```

第 7 章

数据库基础

“Hello World”作为开发的入门示例，已成为习惯。数据库开发的 Hello World 是怎么样的呢？通过本章的学习，读者将能初步掌握数据库开发的基本知识。

数据操作是一个过程，进行具体操作前，先申请数据库资源，操作完成后，必须释放资源。常用的操作主要针对记录，分为增删改查(CRUD)4 种。

CRUD 是数据库技术中的一个缩写词，一般项目开发中各种数据的基本操作都是 CRUD，表示创建(Create)、读取(Read)、更新(Update)和删除(Delete)4 种操作。

CRUD 定义了用于处理数据的基本原子操作。

之所以将 CRUD 提升到一个技术难题的高度，是因为当要完成一个对多个数据库系统同时进行 CRUD 操作的功能时，其性能可能会随数据关系的变化而有非常大的差异。

在开始数据库的学习之前，先阐述一个观点。我们大都已经使用 PC 很久了，然而对于操作系统，有几个人真正了解？绝大多数人只会使用，甚至只会用极小的一部分功能，至于操作系统的原理就更不用说了。我们有必要去了解那么多么！操作系统只是一个工具而已，使用其中的部分功能就可以给我们带来方便。

对于学习，有些东西也不必去了解很深。因为有些东西，如果真要你去说出个所以然来，还真难。比如本章说到的申请数据库资源，为啥要那样写？没法说得很明白。要说的话，只能说这是一个约定俗成的规定，当初就是这样定义的。所以很多东西，我们照着做就行，尤其是初学者，一定要先当一个严格的执行者，照着去做，方能快速入门。等你入门了，就可以去了解更多的东西，否则你只是浪费时间、虚度光阴。

7.1 六万美金项目的核心就一 SQL 语句

周六下午花了 3 个小时，给一大学同学解决了项目中的难题。他带两个小兵，折腾那项目已经两个星期，眼看就要交差了，可核心问题还没解决。

我一到那边，先了解整个项目(费用核销)的需求：从一个数据文件中任选一条内容，

把另一个文件中的相应内容显示出来。

他的解决方式是把两个文件分别读入数组，对相应文件的内容在数组里进行排序。选择后，再一条一条地进行对比，并且还要按不同的条件进行多次对比，这导致速度很慢。记得是有 5 个还是 4 个条件，其中的两个必须满足，记录还得按匹配程度进行排序，即全部符合的排最前，其他依次排列。

我一听，这种方式肯定不行，只能把内容写入数据库，在数据库中通过查询语句来判断。可这条查询语句也很难写，在一个页面里先显示完全满足条件的记录，再显示部分满足的记录，最后显示只和某条件相关的记录。

看他一头雾水，用常规思维确实很难解决。不妨换一种方式，用一个字段来标明：哪些记录是完全满足，哪些是部分满足，哪些是相关，把所有记录合并起来，对那个字段先排序，再按其他要求排序就可以了。当时有 6 种不同组合的条件，常规方式来处理，绝对能难倒绝大部分人。可换一种方式，很快就解决了。看来还是人的思想最重要。

那个项目是一个外包项目，6 万多美金，这钱也太好赚了。

7.2 数据操作种类

先看一个例子：0801.php。注意：请先运行 d:\howwe\30-php-Start.bat，以启动 MySQL 数据库及 PHP 运行环境(可在浏览器中显示 http://127.0.0.1/0801.php)。

参照图 7-1，先在项目 me 上单击鼠标右键，然后选择 **New => PHP File**。在 **File Name** 处输入 0801.php 作为文件名称，然后单击 **Finish**。

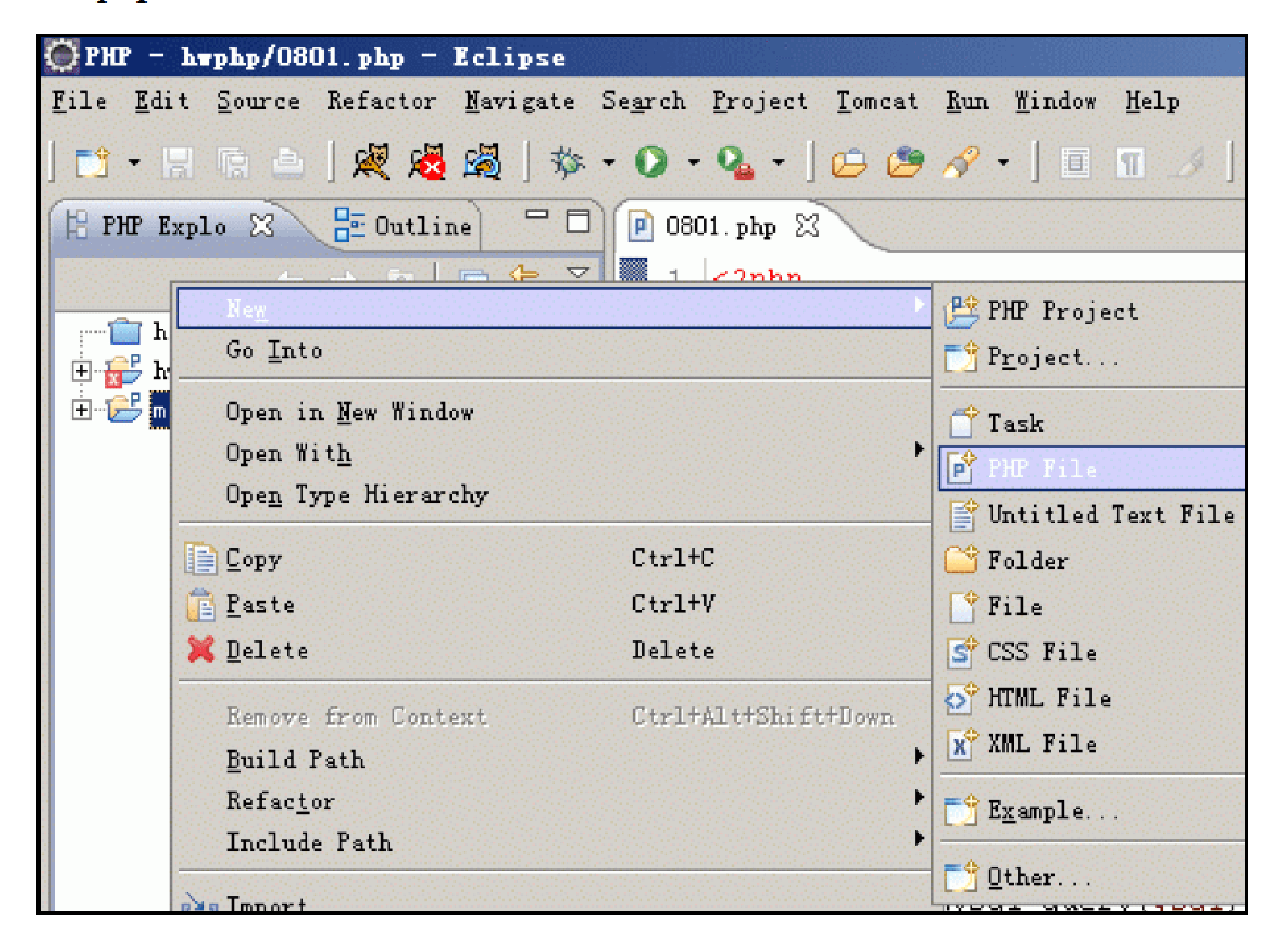


图 7-1 创建 0801.php

在编辑窗口输入图 7-2 所示的代码。图 7-2 仅显示了部分代码，后面将分段详细讲解。请自己查看 hwphp 项目 0801.php，文件具体路径为 D:\howwe\wwwroot\0801.php。

```
<?php
error_reporting(0);
@header("content-Type: text/html; charset=utf-8");

    //创建数据库连接
    $connect = MYSQL_CONNECT( "localhost", "root", "1") or
    mysql_select_db( "a0903171509") or die("Unable to sele
    mysql_query("SET NAMES 'utf8'");// 数据库编码设为utf-8

    //创建表
    $sql="create table test(cnt varchar(200))";
    $result=mysql_query($sql) or die("Can't Create Table".
    echo "创建表: test 成功! \n<br>";

    //增加记录
    $sql="insert into test values('Hello World')";
    $result=mysql_query($sql) or die("Can't Insert");    //
    echo "添加记录\n<br>";
```

图 7-2　创建 0801.php

运行结果如图 7-3 所示。

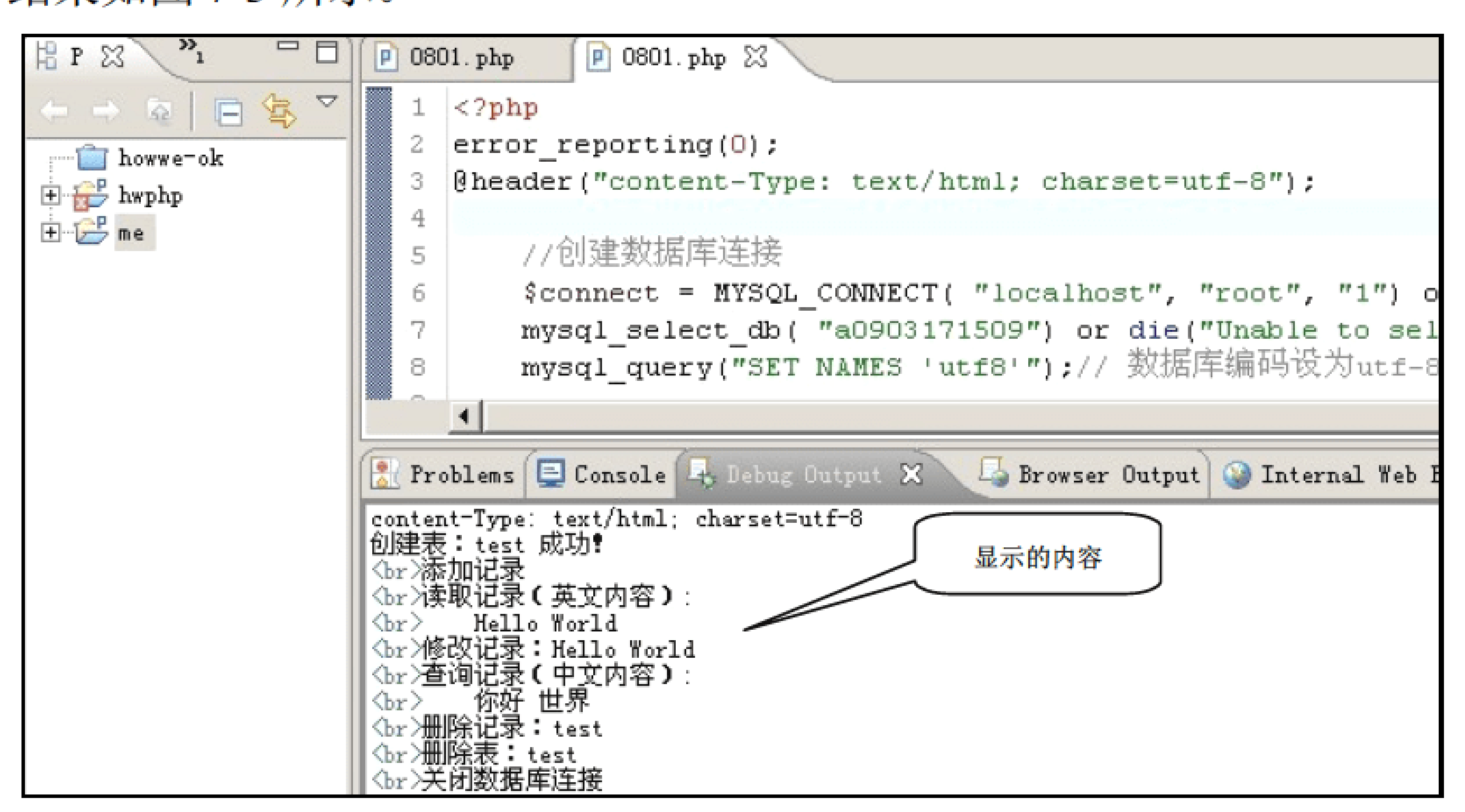


图 7-3　运行结果

初学者一看到这么多代码，估计会头大。但是就这点代码，就完成了创建数据库连接、创建表、增加记录、删除记录、修改记录、查询记录、删除表、关闭记录集、关闭数据库连接等与数据库相关的所有操作。

数据库常用操作有增加记录、删除记录、修改记录、查询记录，创建数据库连接、创建执行语句对象、关闭执行语句对象、关闭数据库连接为基本操作，其余的操作为常用操作所需的相关操作。分类请参见图 7-4。

❖ 注意：

使用 mysql.php 提供的数据库操作指令时，不存在明确的执行语句对象，但标准的数据库访问接口上有，其实 mysql_query 也可以等同于执行语句对象。

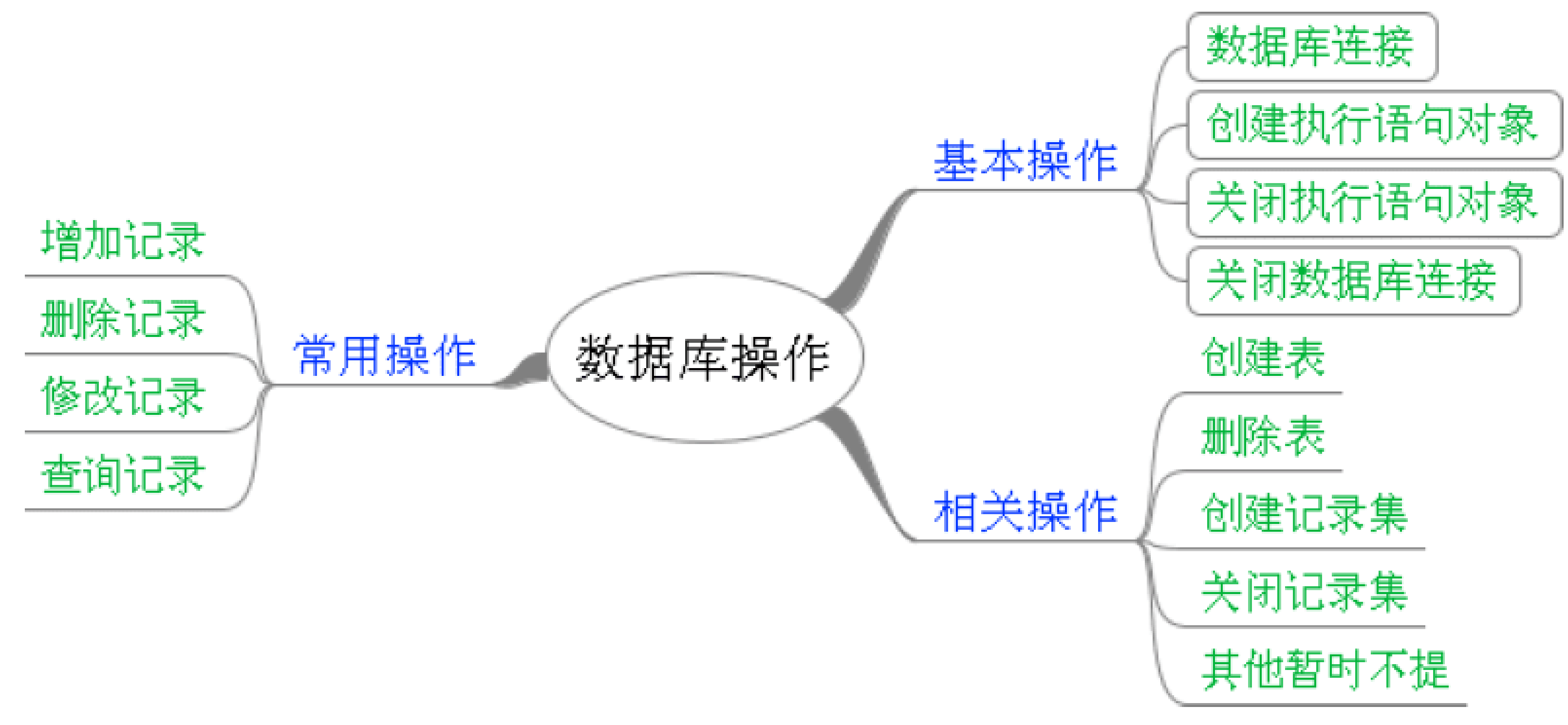


图 7-4 数据库操作的分类

数据的操作过程如图 7-5 所示，常用操作和相关操作只是数据操作中的一个步骤。数据库连接是第一个步骤；创建执行语句对象是第二个步骤；具体操作是第三个步骤，由常用操作或相关操作的一个或多个构成；操作结束，先关闭执行语句对象，再关闭数据库连接。

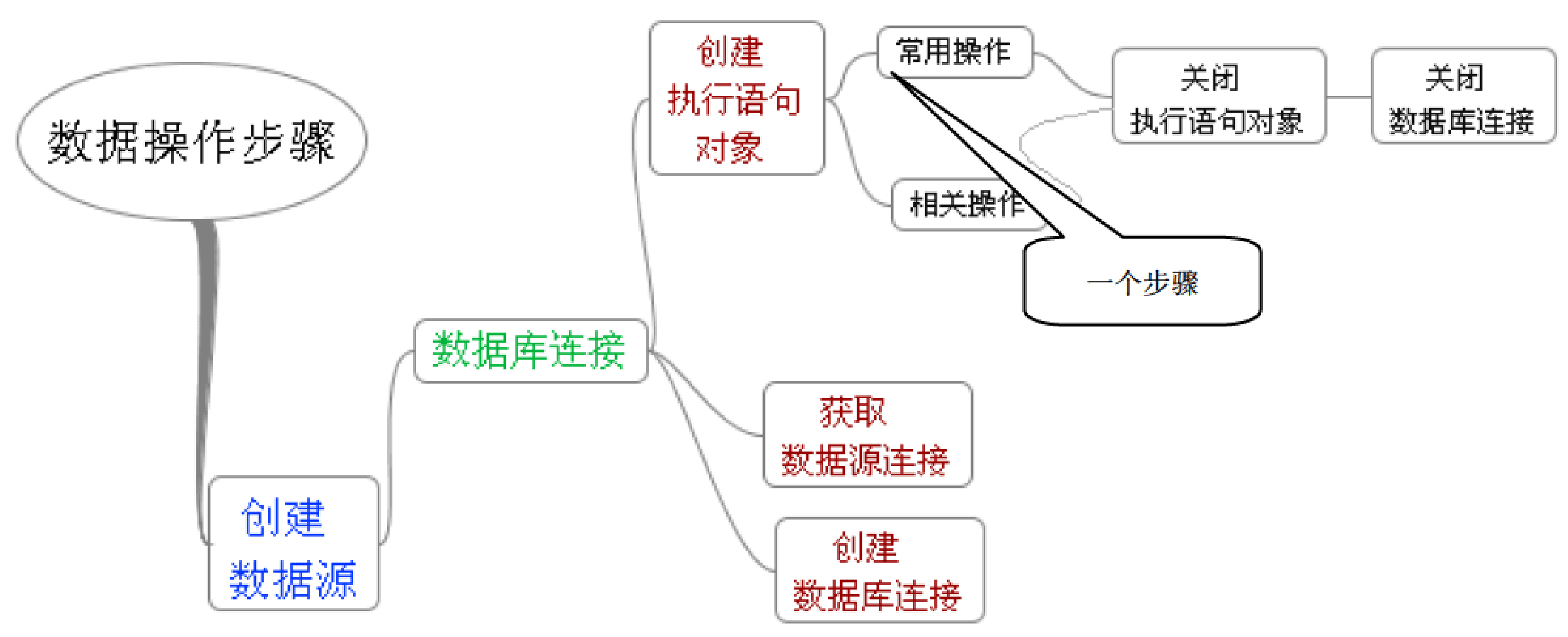


图 7-5 数据的操作过程

其中，数据库连接有两种方式：一种是前面例子所用的创建数据库连接；另一种是获取数据源连接，即从数据库连接池中取得连接。

如图 7-6 所示，在进行数据库的具体操作前，先准备好操作的条件，具体操作完成后，还得恢复原来的状态，所以要关闭执行语句对象和数据库连接。这样才能使操作持续地进行，而不至于耗尽计算机的资源。注意 PHP 脚本一执行完，通常就将自动释放所有资源，但为了程序的完整性，建议加上释放数据库资源的代码。



图 7-6 数据操作的具体步骤

通俗一点讲，不论对计算机进行何种操作，我们都先要准备资源，操作完成后，必须

释放资源。如果不释放，资源只会越用越少，并最终使资源耗尽。

日常使用的页面中，一般只有一至两种操作，所以单纯写点数据库操作方面的代码很容易，唯一难的是要能整体掌握项目的需求。故本书一直看重整体思维，而不只是一个一个的知识点。而这些知识点，用 Google 一搜一大把，同时本书还将整理一系列的知识点，发布在网站 howwe.net 上。

本书不会讲述很多详细的知识点，只会提出一些能将这些知识点串起来的过程，具体的知识点请读者自己去搜索。

7.3　申请资源

申请资源，是进行具体数据操作的前提。主要步骤如下：

(1) 建立数据库连接。

(2) 创建执行语句对象。

代码如下：

```
error reporting(0);
@header("content-Type: text/html; charset=utf-8");

//创建数据库连接
$connect = MYSQL CONNECT( "localhost", "root", "1") or die("Unable to connect
to MySQL server");
mysql select db( "a0903171509") or die("Unable to select database");
mysql_query("SET NAMES 'utf8'");// 数据库编码设为 utf-8
```

这里主要是采用 mysql.php 提供的方法。mysql.php 用于 PHP 语言访问 MySQL 数据库，其中常用的方法如下：

mysql_connect：打开一个到 MySQL 服务器的连接，一旦脚本结束，到服务器的连接就会被关闭，也可调用 mysql_close()进行主动关闭。若使用同样的参数进行第二次调用，则不会建立新连接，而是返回已经打开的连接标识，除非使用参数 new_link 要求打开新连接。

mysql_query：发送一条 MySQL 语句。

mysql_result：取得结果数据。

mysql_fetch_array：从结果集中取得一行作为关联数组或数字数组，或二者兼有。

mysql_pconnect：打开一个到 MySQL 服务器的持久连接。

7.4　常用数据操作

常用数据操作主要用于操作记录，简单一点说，就 4 个字——增删改查(CRUD)。

- 增：创建/增加，Create/Insert。
- 删：删除，Delete。
- 改：修改，Update。
- 查：读取/查询，Read/Select。

CRUD 定义了用于处理数据的基本原子操作。

之所以将 CRUD 提升到一个技术难题的高度，是因为当要完成一个对多个数据库系统同时进行 CRUD 操作的功能时，其性能可能会随数据关系的变化而有非常大的差异。

```
//增加记录
$sql="insert into test values('Hello World')";
$result=mysql query($sql) or die("Can't Insert");//执行给定的 SQL 语句
echo "添加记录\n<br>";

//查询记录
$sql="SELECT cnt FROM test";
$result=mysql_query($sql) or die("Can't Read");  //执行给定的 SQL 语句
echo "读取记录(英文内容):\n<br>";
while($rs=mysql_fetch_array($result)) {
    echo "\t".$rs['cnt']."\n<br>";
}

//修改记录
$sql="update test set cnt='你好 世界' where cnt='Hello World'";
$result=mysql_query($sql) or die("Can't Update");//执行给定的 SQL 语句
echo "修改记录：Hello World\n<br>";

//查询记录
$sql="SELECT cnt FROM test";
$result=mysql_query($sql) or die("Can't Read");  //执行给定的 SQL 语句
echo "查询记录(中文内容):\n<br>";
while($rs=mysql_fetch_array($result)) {
    echo "\t".$rs['cnt']."\n<br>";
}
mysql_free_result($result);//关闭记录集

//删除记录
$sql="delete from test";
```

```
    $result=mysql_query($sql) or die("Can't delete Table");   //执行给定的
                                                                SQL 语句
    echo "删除记录：test\n<br>";
```

7.5　数据库模型

常用数据操作之外的其他操作称为相关数据操作。数据库有如下对象：数据库、表、视图、存储过程、记录、字段、索引、键，除了记录之外，其他对象也需要增删改查，相关数据操作就是对这些对象进行处理。

在此不再深入讲述，只留下一个整体概念，数据库对象如图 7-7 所示，数据库模型如图 7-8 所示。

数据库原理，有些书讲述了一堆又一堆的理论，一看就让人头痛。但对绝大多数人来说，简单概括如图 7-8 所示，数据库就分为数据库对象和数据库操作，大多数据库应用只是使用 SQL 语句操作数据库对象而已。而其他知识，除非要去进行这方面的研究，没有多大必要去浪费时间。

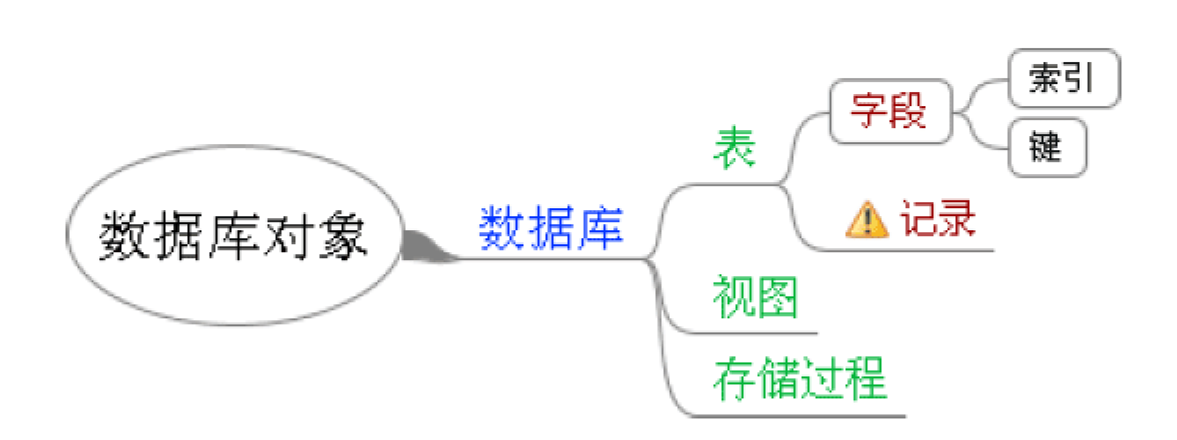


图 7-7　数据库对象

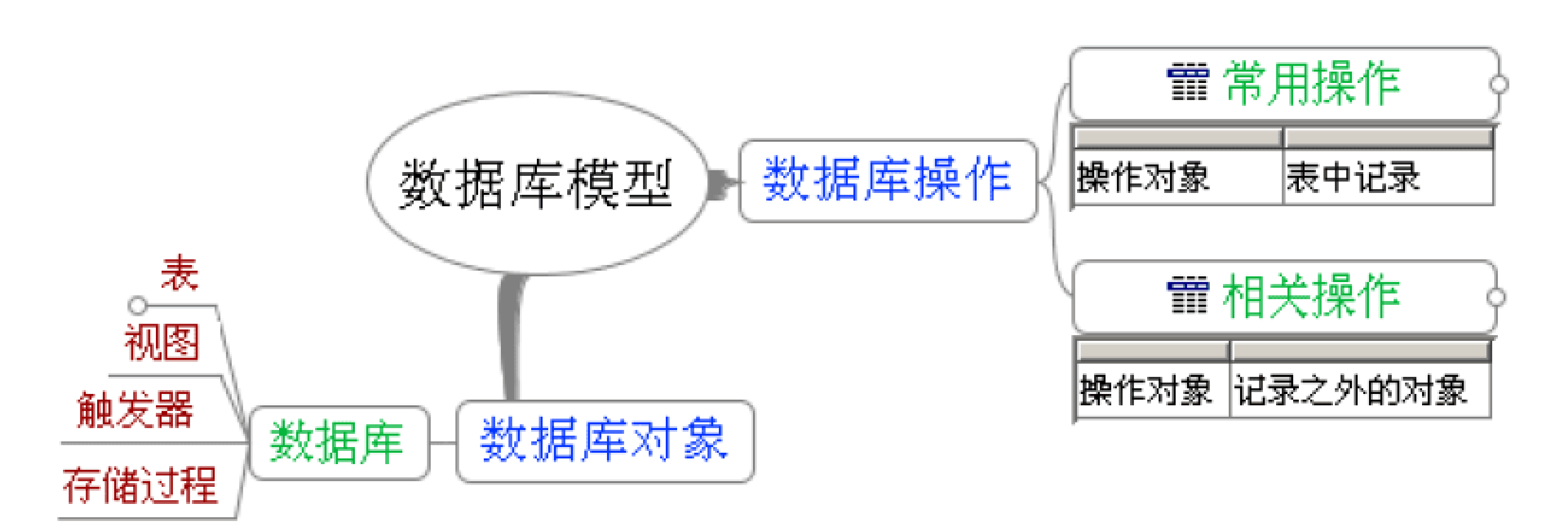


图 7-8　数据库模型

```
//创建表
$sql="create table test(cnt varchar(200))";
$result=mysql_query($sql) or die("Can't Create Table".mysql_error());
//执行给定的 SQL 语句
    echo "创建表：test 成功！\n<br>";
```

```
    //删除表
$sql="drop table test";
$result=mysql_query($sql) or die("Can't drop Table");//执行给定的 SQL 语句
    echo "删除表：test\n<br>";
```

7.6 释 放 资 源

具体操作完成以后，必须释放资源，使其可以充分利用，而不至于耗尽资源。

```
mysql_close($connect);   // 关闭数据库连接
echo "关闭数据库连接";
```

7.7 小语句解决大难题，IT 需要简单化

一朋友碰到一个难题，请我帮他写个小程序。要求如下：

有一个 Excel 文档，三张表：NBS(近 7 千条记录)、Sheet1、CUSTOM(2 万条记录)。里面内容是国内一些企业的资料，要求找出 NBS 和 CUSTOM 两张表里同时出现的企业，即找出名称相同的企业，然后将企业名称和各自的企业代码复制到 Sheet1 里面。

那朋友先是手工操作，找了 200 多条，估计花了很多时间。但因为实在太多，只得请求帮助。

做法如下：选中工作表 Sheet1，从菜单中选择：**工具 -> 宏 -> Visual Basic 编辑器**，双击 Name 为“Sheet1”的对象，出现一个编辑窗口，如图 7-9 所示，椭圆选择区为双击的地方，方框选择区为代码区。下载地址：hwcall.googlecode.com/files/Howwe4-6.6.rar。其中表“Sheet2”为示范工作表，NBS 和 CUSTOM 仅保留两栏信息。

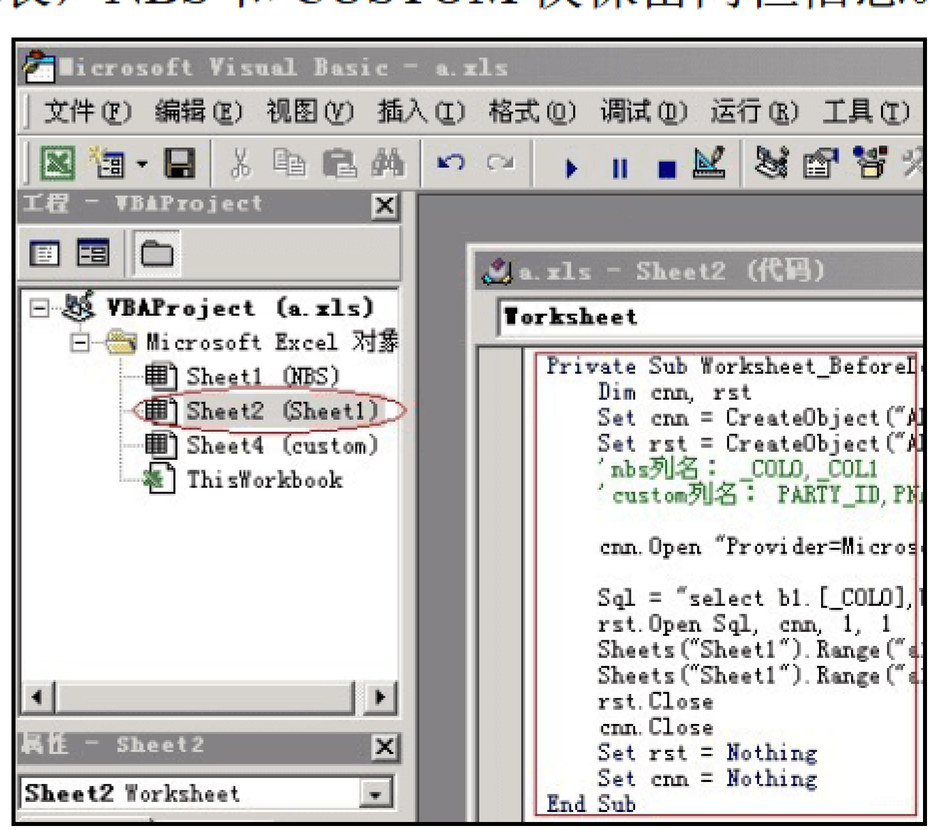


图 7-9 Excel 快速操作

copy 以下代码，保存后关闭，任意双击 Sheet1 的一个单元格即可。

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As
                                       Boolean)
    Dim cnn, rst
    Set cnn = CreateObject("ADODB.connection")
    Set rst = CreateObject("ADODB.recordset")
    //nbs 列名:  _COL0,_COL1
    //custom 列名:  PARTY_ID,PNAME

    cnn.Open "Provider=Microsoft.Jet.OLEDB.4.0;
              Extended Properties='Excel 8.0;';
              Data Source=" & ThisWorkbook.FullName

    Sql = "select b1.[_COL0],b2.PARTY_ID,b1.[_COL1] from [nbs$] as b1,
           [custom$] as b2 where b1.[_COL1]=b2.PNAME"
    rst.Open Sql, cnn, 1, 1
    Sheets("Sheet1").Range("a1:c10000").ClearContents   //Sheet2 为测试用的
                                                          工作表，换成自己的
    Sheets("Sheet1").Range("a1").CopyFromRecordset rst
    rst.Close
    cnn.Close
    Set rst = Nothing
    Set cnn = Nothing
End Sub
```

也许很多人不懂 VBA，但这是最简单的解决方式。其实用什么语言来编程，不是最重要的。语言只是问题解决的体现，是你解决问题的一种方式而已。所以我们更需要的是培养自己的思维，培养自己解决问题的能力。

拿以上的 VBA 来说，除了：`Sql="select b1.[_COL0],b2.PARTY_ID,b1.[_COL1] from [nbs$] as b1,[custom$] as b2 where b1.[_COL1]=b2.PNAME"`之外，其他都是固定的，你也不用去了解为啥要这样写。就如同使用操作系统一样，只要会用，而根本不需要明白操作系统为啥能那样用。可以问问自己，除了能使用之外，还了解多少操作系统的知识。道理一样，编程时，我们很多时候只要知道用什么能解决问题即可。

❖ **提示：**

[_COL0]指标题为“_COL0”的那一列，如果没有标题，就用 F1 代表第一列，其他依此类推。

就拿我自己来说，很少用 VBA，还是 1999 年才用过半年的 VB，但我知道用 VBA 能解决问题，于是去 Google 资料，绝大多数人用的是 vlookup。7000 条数据，如果用 vlookup，

需要很长时间，所以我就查 SQL 的使用方式。

这种方式看起来代码不少，但很多代码你根本不用去管，照样 Copy 过来，改一下你要查什么内容，即修改一下 SQL 的定义(这条 SQL 语句的意思很简单：select 你要的东西，from 你要获取数据的地方，where 你需要查看的范围。不要觉着 SQL 语句高不可攀，其实这就和你想问题的方式一样)，再改一下你要把数据放在哪里就行了。

IT 需要简单化，上面讲的方法很容易掌握，我相信你以后能用它解决更多问题。

小知识：注意编程风格

1) 在编写程序时，可以在运算符的左右或是逗号“,”之后适当地使用一些空格，让程序看起来不是那么拥挤，不仅程序比较美观，阅读起来也比较容易，比较一下就可以有所体会：

```
int i=0;
int number=0;
number=i++;
number=i--;
```

下面的写法会比较好读一些：

```
int i = 0;
int number = 0;
number = i++;
number = i--;
```

2) 在编写程序时适当地使用缩进(Indent)，可以在定义程序块时让程序块范围看着容易分辨。每个开发人员使用缩进的方式各不相同，你可以使用两个空格或是 4 个空格，也可以按下 Tab 键来进行缩进。但各个文字编辑器或 IDE 对于 Tab 字符的显示方式都不太一样。

有些人的习惯：缩进方式是按 4 次空格键，不按 Tab 键，因为各个文字编辑器或 IDE 对于 Tab 字符的显示方式都不太一样。有些文字编辑器或 IDE 默认使用 Tab 字符来自动缩进，可以将这改为默认按 4 个空格符进行缩进，因为空格符的显示方式是一致的。当然缺点就是：若要改变程序排列，就要不断地按空格键。

第 8 章 SQL 基础及数据库管理

很多人觉得 SQL 高不可攀，其实从本质来说整个 SQL 可以简化为 4 句话：insert、delete、update、select，增删改查 4 字而已。

数据库管理主要是针对数据库中的对象进行管理，数据库对象分为数据库、表、记录、存储过程、函数等。总的来说，数据库可简单归纳为一个模型，分为数据库对象和数据库操作，SQL 语句就是针对数据库对象的操作语言，大多应用程序只是采用 SQL 操作数据库对象而已。

8.1 SQL 基础

SQL是用于访问和处理数据库的标准计算机语言。

正如前面所说，数据库就是图8-1所示的一个模型，分为数据库对象和数据库操作，SQL语句就是针对数据库对象的操作语言。

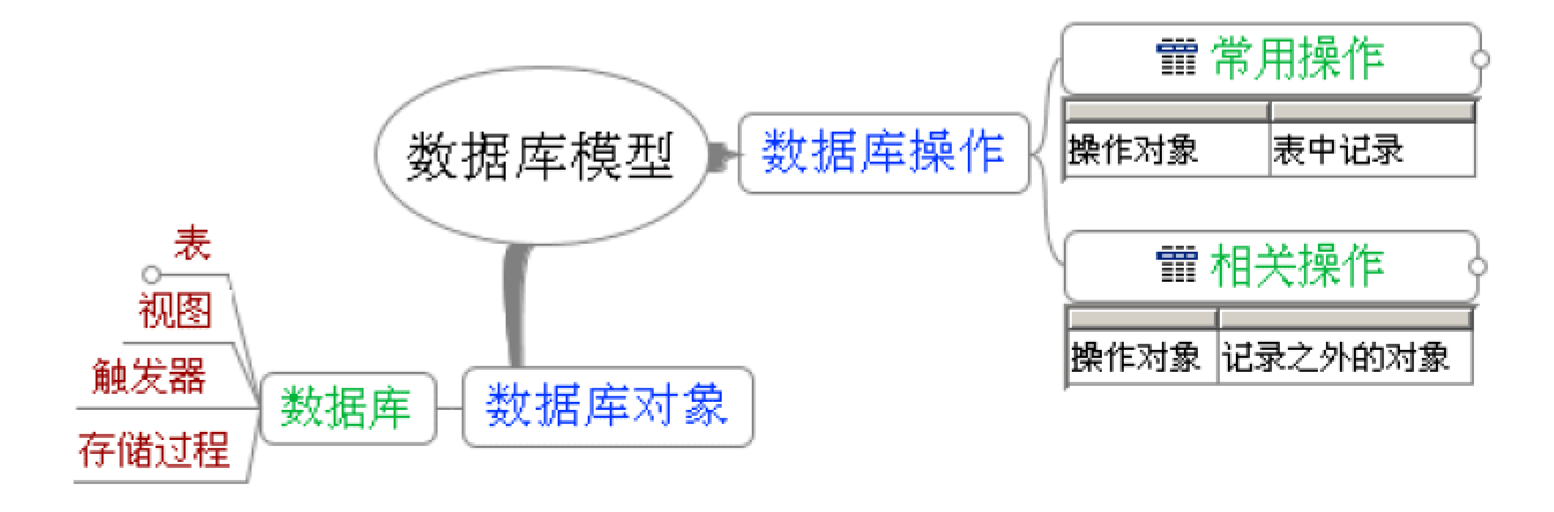


图 8-1 数据库模型

SQL 是 ANSI(美国国家标准化组织)的一种标准计算机语言，是使我们有能力访问数据库的结构化查询语言。

SQL 能做什么？简单一点说就是能对数据库对象进行处理。

- 可从数据库查询出数据——select
- 可在数据库中插入新的记录——insert
- 可更新数据库中的数据——update
- 可从数据库删除记录——delete
- 可操作数据库——create/alert/drop database
- 可在数据库中操作表——create/alert/drop table
- 可在数据库中操作索引——create/alert/drop index
- 可在数据库中操作存储过程——create/alert/drop procedure
- 可在数据库中操作视图——create/alert/drop view
- 可在数据库中操作触发器——create/alert/drop trigger
- 可以设置表、存储过程和视图的权限——grant/deny/revoke

其中常用的就 4 句：select、insert、update、delete，其余语句一般只有数据库管理员才用。

❖ **提示：**

每种数据库系统都有自己的 SQL 数据类型及控制结构，具体请参考该数据库的帮助信息。

8.2 数据操作-针对记录

数据操纵语言(Data Manipulation Language，DML)包括 4 条语句：select、insert、update、delete。

1. 查询(select)

简单 SELECT 语句的语法如下：

```
SELECT [ALL|DISTINCT] <目标表达式>[, <目标表达式>]...
FROM <表或视图名>[, <表或视图名>]...
[WHERE <条件表达式>]
[GROUP BY <列名 1> [HAVING <条件表达式>]]
[ORDER BY <列名 2> [ASC | DESC] ]
```

含义：

根据 WHERE 子句的条件表达式，从 FROM 子句指定的表或视图中找出满足条件的元素组，再按 SELECT 子句中的目标列表达式选出元素组中的属性值，形成结果表。

如果有 GROUP 子句，就将结果按<列名 1>的值进行分组，该属性值相等的元素为一个组，每个组产生结果表中的一条记录。如果 GROUP 子句带有 HAVING 短语，则只有

满足指定条件的组才予以输出。

如果有 ORDER 子句，则结果表还要按<列名 2>的值做升序或降序排序。

查询有连接查询、子查询(也叫嵌套查询)、合并查询(union)等几种。

如果一个查询需要对多个表进行操作，就称为连接查询。查询主要通过各个表之间共同列的关联性来查询数据，它是关系数据库查询最主要的特征。

连接查询分为等值连接查询(连接条件使用=)、非等值连接查询(连接条件使用>、>=、<、<=、!=及 BETWEEN…AND)、自连接查询(表使用 Join，同其自身进行连接)、外连接查询和复合条件连接查询(WHERE 子句中使用多个连接条件，简称为复合查询)。

外连接查询有两种：主表在左边，为左外部连接，简称左连接(Left Join)；主表在右边，为右外部连接，简称右连接(Right Join)。

❖ **注意：**

可以将 select 出来的结果生成一个新表，方法是在 from 前面加“into 新表名 ”。

2. 插入(insert)

插入单条记录：

```
INSERT INTO 表名称 VALUES (值1, 值2,...)
```

指定要插入数据的列：

```
INSERT INTO table_name (列1, 列2,...) VALUES (值1, 值2,...)
```

增加多条记录：插入子查询结果或存储过程返回的结果。

3. 修改(update)

修改单列数据：

```
UPDATE 表名称 SET 列名称 = 新值 WHERE 列名称 = 某值
```

也可以修改多列数据：

```
UPDATE table_name SET 列1=值1, 列2=值2,... WHERE 列名称 = 某值
```

4. 删除(delete)

语法如下：

```
DELETE FROM 表名称 WHERE 列名称 = 值
```

8.3 数据定义-针对对象

数据定义语言(Data Definition Language，DDL)主要是针对数据库对象进行操作。这些数据库对象包括数据库(database)、表(table)、索引(index)、视图(view)、存储过程(procedure)、触发器(trigger)等，操作一般有创建(create)、修改(alert)、删除(delete)三种。

存储过程的执行一般使用 EXECUTE。详细情况请自己 Google。

8.4 数据控制-授权

数据控制语言(Data Control Language，DCL)主要是管理用户所能进行的具体操作，一般用户很少使用。具体情况请参考使用的数据库软件，本书不再展开。

8.5 MySQL 快速入门

如果想更详细地了解 MySQL，请自己去 Google，本节仅简单说明 MySQL 的基本用法，以便读者快速上手。

❖ 提示：

浩为开发包中有 MySQL 主程序及管理程序，目录分别为 D:\howwe\mysql 和 D:\howwe\mysql\sqlfont。如使用浩为开发包，可略过以下的下载和安装内容部分。

1. 下载

从以下地址可以下载绿色版的 MySQL(此版本安装十分方便，很适合初学者)：http://zyt.howwe.net/me.php?154。再下载图形管理工具，如图 8-2 所示。



图 8-2 下载 MySQL

2. 安装

1) 将 MySQL 5.1 绿色版解压到 D 盘，解压后的目录如图 8-3 所示。如果不一致，请调整。

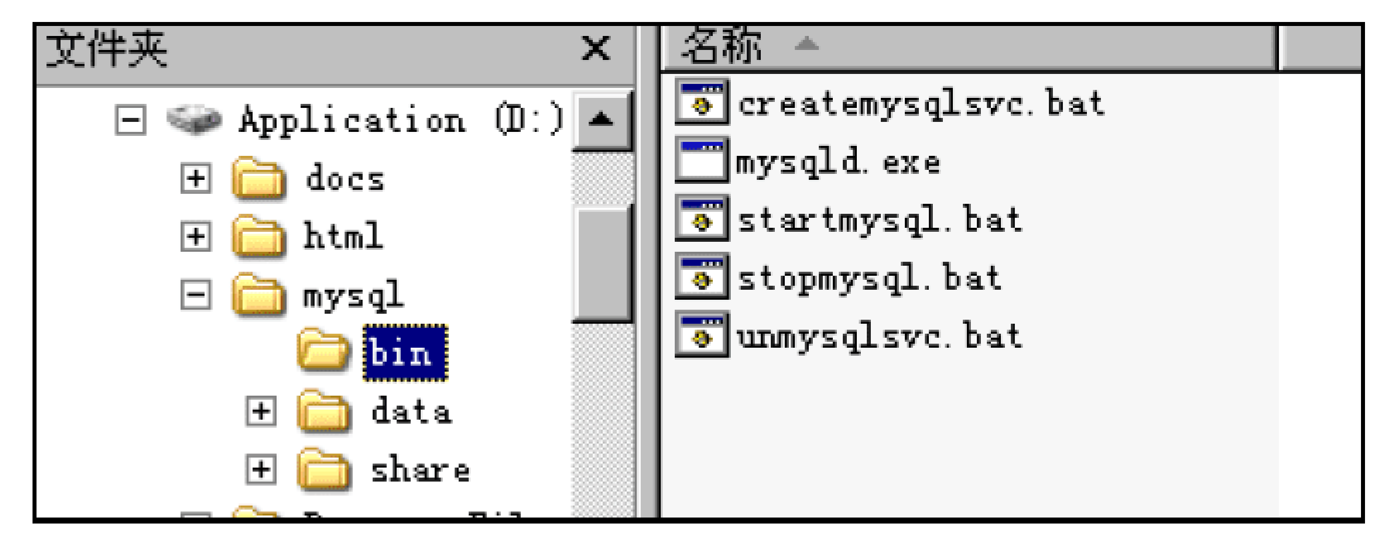


图 8-3　MySQL 目录

绿色版的数据文件默认地址为 D:\mysql，如需更改，请修改 mysql 目录下 my.ini 文件的 basedir 和 datadir 两项或注销这两项(注销后将默认为当前目录)，注意 MySQL 只支持绝对路径。

要运行 MySQL，先双击 createmysqlsvc.bat 将 MySQL 注册成服务，计算机重启后它将自动启动，再双击 startmysql.bat 即可启动服务。要停止服务，请双击 stopmysql.bat。如果不需要 MySQL 随着计算机的启动而自动运行，请运行 unmysqlsvc.bat。

2) 安装管理界面。解压 mysql-front，解压后的文件如图 8-4 所示。

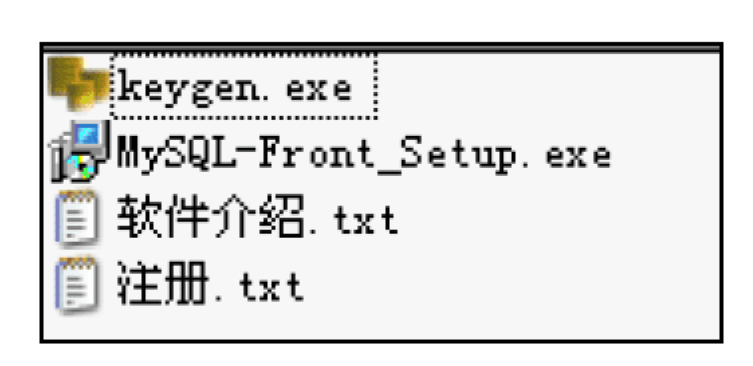


图 8-4　mysql-front 解压后的文件

双击 MySQL-Front_Setup.exe，安装时，一路默认即可。

3. 运行管理界面，并连接到 MySQL

1) 启动 MySQL 服务，双击图标运行，第一次运行时，会弹出两个对话框，直接单击**取消**。

接着选择菜单里的 **帮助** -> **登记** -> **要求输入 Key**，此时输入注册码(在文件“注册.txt”中)即可。

2) 选择**文件->打开登录信息**，如图 8-5 所示。

弹出**添加信息**对话框，进行图 8-6~图 8-8 所示的设置。

注册：用户名 root，密码 1。

连接：服务器 localhost，字符集 utf8，其他默认。



图 8-5　打开登录信息

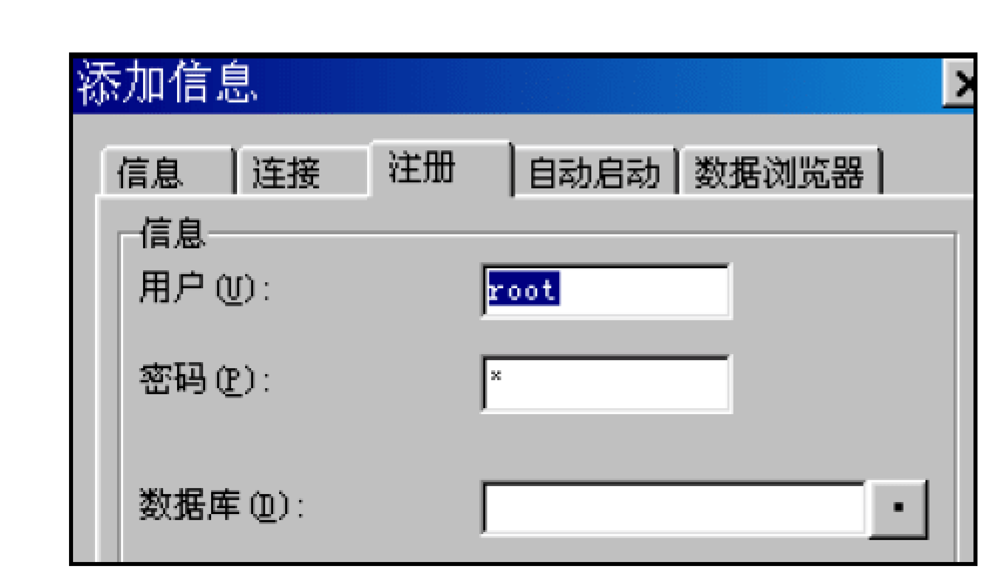


图 8-6　设置注册信息



图 8-7　设置连接信息



图 8-8　查看设置信息

单击**确定**，弹出打开登录信息的对话框，单击**打开**(参见图 8-9)，进入图 8-10 所示的 MySQL-Front 管理界面。



图 8-9　“打开登录信息”对话框

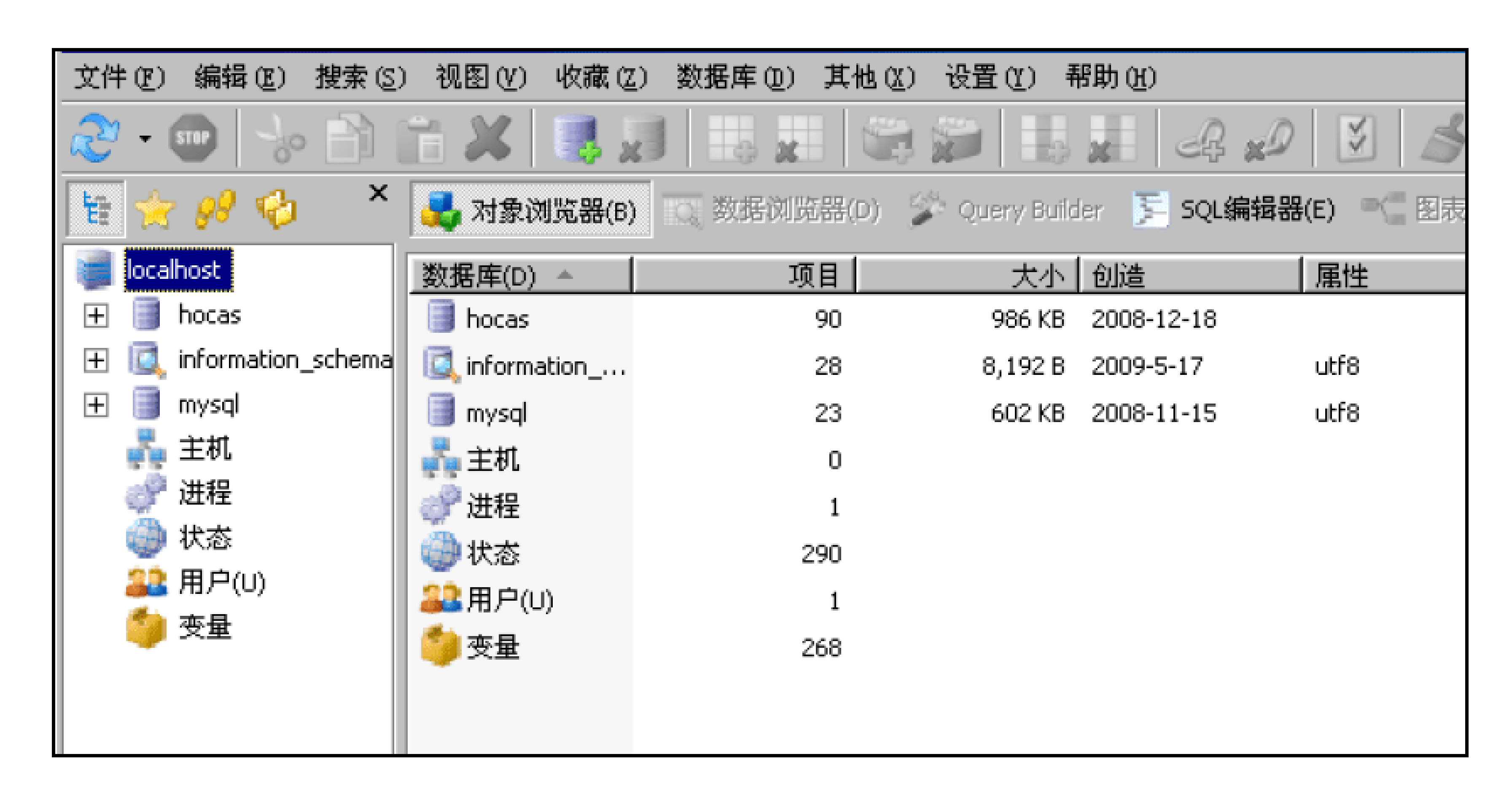


图 8-10　MySQL-Front 管理界面

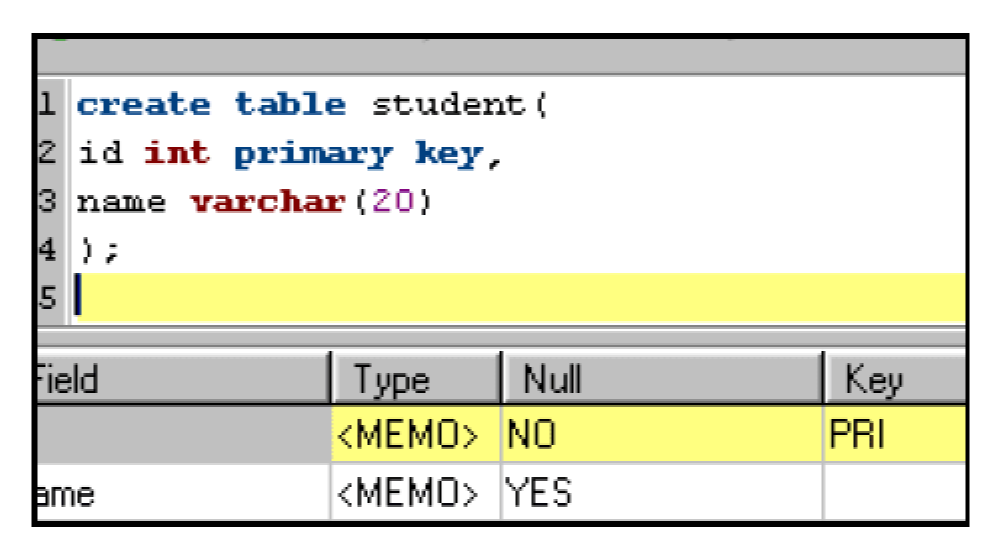
此时，已成功连接至 MySQL 服务器。

4. 使用 SQL 进行基本操作

单击 **SQL 编辑器**(参见图 8-11)，得到图 8-12 所示的界面。



图 8-11　单击 **SQL 编辑器**




```
create table idcard(
id int primary key,
studentid int ,
cardInfo varchar(50) not null,
foreign key(studentid) references student(id)
)
```

Field	Type	Null	Key	Default	Extra
d	<MEMO>	NO	PRI		
tudentid	<MEMO>	YES	MUL		
ardInfo	<MEMO>	NO			

图 8-12　创建数据表

1) 创建数据库

```
create database test;
```

2) 选择使用该数据库

```
use test;
```

3) 建表

student 表：

```
create table student(
id int primary key,
name varchar(20)
);
```

一个学生，一张 IdCard。

idcard 表：

```
create table idcard(
id int primary key,
studentid int ,
cardInfo varchar(50) not null,
foreign key(studentid) references student(id)
)
```

4) 增删改查操作(效果见图 8-13、图 8-14 和图 8-15)

增：

```
insert into student  values(0,'ad');
insert into student values(1,'ad1');
```

查：

```
    select * from student;
```

改：

```
update student set name = 'ad_update'  where id = 1
```

删：

```
delete from student where id = 1
```

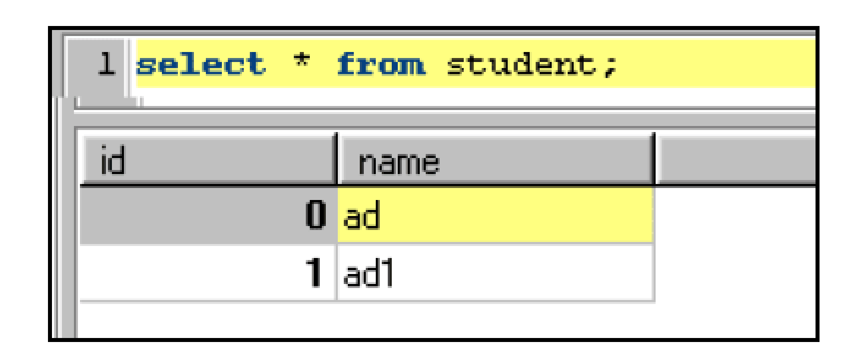


图 8-13 插入数据并查询

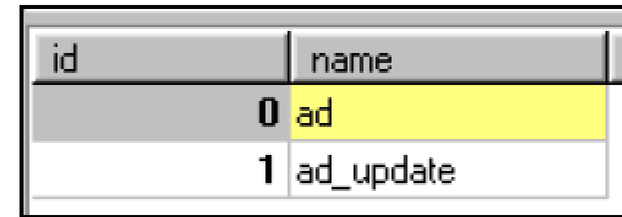


图 8-14 修改数据



图 8-15 删除数据

5) MySQL 所支持的自动递增功能

一个表只能有一个 AUTO_INCREMENT 属性，且该属性必须为主键的一部分。

```
create table test1(id int primary key AUTO_INCREMENT,name varchar(20));
```

6) MySQL 所支持的分页处理功能

先在 student 表中添加数据：

```
insert into student values(1,'ad1');
insert into student values(2,'ad2');
insert into student values(3,'ad3');
insert into student values(4,'ad4');
insert into student values(5,'ad5');
```

查看表中的数据以对比，如图 8-16 所示。

使用 limit 关键字：

```
select * from student limit 2,3;
```

上述查询从第 2 条记录开始，取出 3 条数据，结果如图 8-17 所示。

❖ 注意：

数据库中的记录是从 0 开始计算的。

id	name
0	ad
1	ad1
2	ad2
3	ad3
4	ad4
5	ad5

图 8-16　查看表中的数据

id	name
2	ad2
3	ad3
4	ad4

图 8-17　从表中取出 3 条记录

8.6　sqlFont 及数据库管理软件

sqlFont 是 MySQL-Front 的升级版，仅用于对 MySQL 的管理。MySQL-Front 的使用在 **8.5 MySQL 快速入门**中已有介绍，可参考其使用方法，在此不再详细介绍，仅留其主窗口截图，如图 8-18 所示。

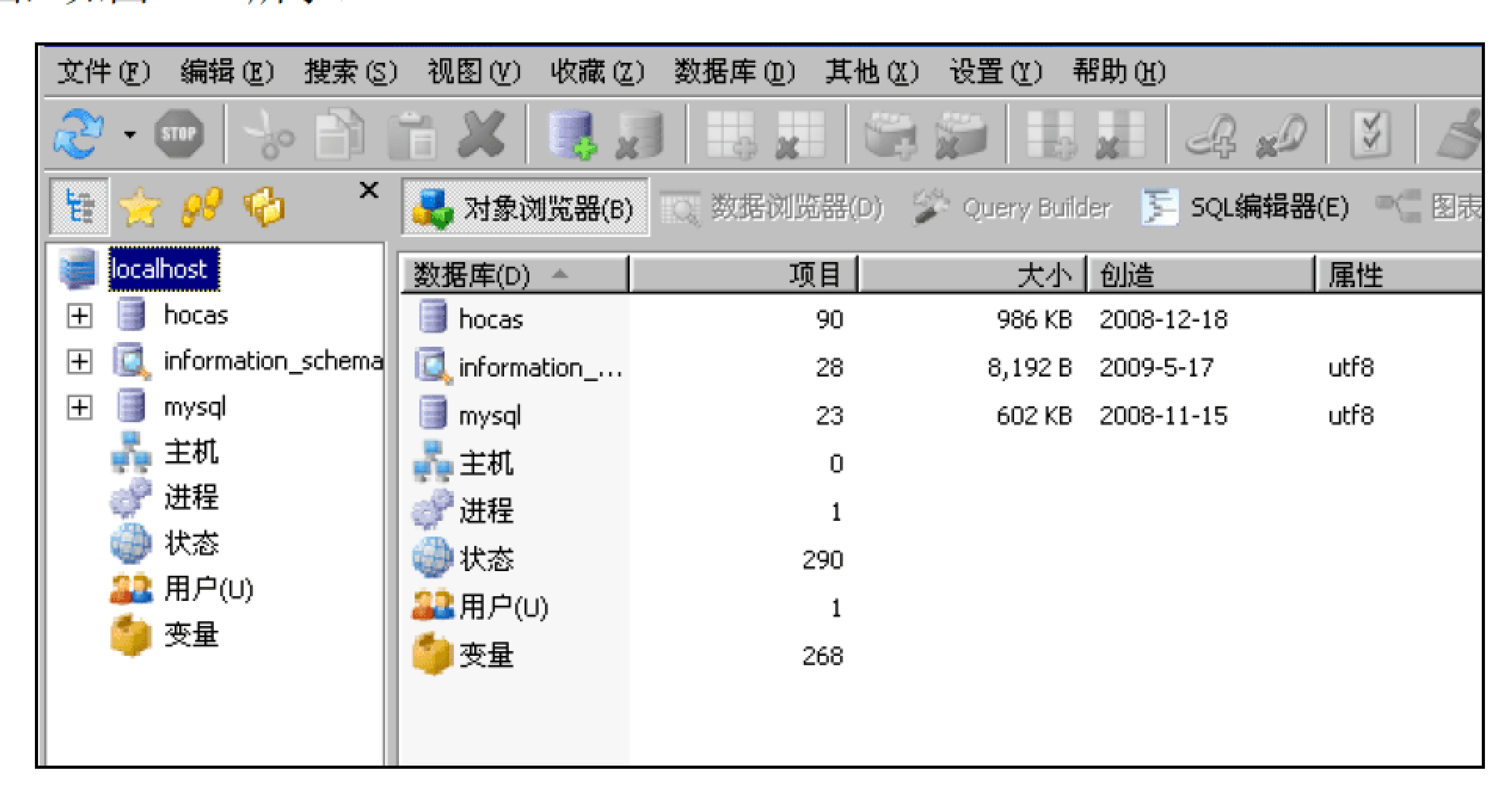


图 8-18　sqlFront 主窗口

SQLExplorer 是 Eclipse 的一个插件，可用来管理各种数据库。SQLExplorer 提供了一个使用 SQL 语句访问数据库的图形用户接口(GUI)。通过使用 SQLExplorer，能够显示表、表结构和表中数据，并提取、添加、更新或删除表数据。SQLExplorer 同样能够生成 SQL 脚本来创建和查询表。所以，与命令行客户端相比，使用 SQLExplorer 是更优越的选择。

通过在 Eclipse 中加入一个 SQLExplorer 视窗，SQLExplorer 为 Eclipse 配置了一个访问数据库的 SQL 客户端。

SQLExplorer 的当前版本为 3.5，有两个版本：Eclipse 插件和独立客户端(Standalone Client)。其中，独立客户端是一 RCP 程序，可以直接使用，不用依赖于 Eclipse。下载地址为 http://eclipsesql.sourceforge.net/。

通过在 Drivers 标签中为数据库选择合适的驱动，可以配置与其他数据库的 JDBC 连接。只需为选定的数据库指定驱动类和连接 URL，就可以配置与该数据库的 JDBC 连接。

SQLExplorer 的使用详情请参考《Java 快速入门与商用项目培训》的“8.2 配置 SQLExplorer”和“8.3 使用 SQLExplorer”，也可以自行 Google。

因为这两个软件包含在浩为开发包中，故这里只作简单介绍，其实每种数据库都有自带的数据库管理软件。记住，这些管理软件只是工具而已，不用花费太多的精力，一般只要多试试就会用。

第 9 章

PHP 开发基础

PHP 很快就能上手，但除了掌握基本的 PHP 语法之外，还得掌握和 PHP 相关的内容。但到底该掌握哪些内容，又该掌握到什么程度？很多人很迷茫。学以致用是解决问题的关键，通过本章的学习，你将不再迷茫，本章把相关知识点串成一条线，从而使你快速步入开发之门。

对于语法的学习，一般只需要先了解有哪些基本的语法，然后在使用中实践并验证这些语法，并不断学习新的语法即可。切不可在完全搞懂语法后才去实践，这样只会浪费时间。

总之，你得尽快从一个又一个的页面先做起来，哪怕是最简单的页面。当你看到页面的效果时，自豪感便会油然而生……

这样，你很快就入门了。

9.1 PHP 的基本特征

PHP 的特征在“1.1 PHP 特征：请求无法持久”中已有介绍。总之，PHP 是一种专门针对 Web 开发设计的语言，对 Web 常见的需求都有内置的实现；而且 PHP 是一种**“每次请求就是一个完整的生命周期”**的语言，每请求一次，所有相关对象都必须重新加载，这些对象在请求一结束就自动销毁，这样就彻底杜绝了内存和资源的泄露。PHP 另一个特点是页面创建的变量和其他对象，都只在当前页面的内部可见，无法跨越页面访问。

❖ **注意：**

为了留下更深的印象，请读者对本节提到的内容反复阅读，仔细体会体会，以真正了解 PHP 的特征，这样在 PHP 的开发中便能得心应手。

9.2 HTML 语 法

HTML 是 HyperText Mark-up Language 的英文缩写，即超文本标记语言，是一种用来制作网页的标记语言，不需要编译，可直接由浏览器执行。对大小写不敏感，HTML 与 html 是一样的，由 W3C 维护，最新版本为 HTML 5。

HTML 文件是一个文本文件，包含了一些 HTML 元素、标签等。HTML 文件必须使用 html 或 htm 作为文件名后缀。

1. HTML 的文件框架

文件框架的作用是描述 HTML 文件的结构，但 HTML 中有一种元素就叫框架(Frame)。

例 1001. htm：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html>
     <head>
       <title>HTML 文件标题</title>
       <!-- HTML 头信息 -->
     </head>

     <body>
       HTML 内容信息
     </body>
</html>
```

HTML 文档首先要声明一个文档类型，也就是例 1001.htm 中的第一行，定义了 XHTML 1.1 文档类型(文档类型是为了说明这个页面使用了何种 HTML 规则或结构，一般可以省略)。

<html>和</html>是 HTML 文档的开始与结束标记，也是 HTML 文档的根元素。除了文档类型以外的所有页面内容都包括在 html 元素内。

HTML 文件主要分为头信息 head 与内容信息 body：

- head 信息：头信息 head 可以容纳文档的 HTML 相关信息，比如标题 title、页面的语言与文字类型、CSS 样式、JavaScript 代码、简短描述、关键词等内容，是用户无法直接看到的。
- body 信息：内容信息 body 包括用户可以看到的全部内容，比如段落、链接、表格等。

❖ **注意：**

编写代码时一定要有层次感，可使用 Tab 键来产生 HTML 代码的层次感，这样可以使代码更清晰，更容易扩展。不过 Tab 键在不同编辑器上占用的空格数不同，有的为 4 个，有的为 8 个，建议用 4 个空格代替 Tab 键。

2. 常用的 HTML 标签

1) HTML 框架类标签

- html：定义 HTML 文档。
- body：定义文档内容信息。
- head：定义文档头信息。
- title：定义文档的标题。
- <!-- ... -->：HTML 注释标签。

2) HTML 图像与链接类标签

- a：HTML 链接标签。
- img：HTML 图像标签。

3) HTML 文字相关标签

- h：定义标题 1 至标题 6——h1、h2、h3、h4、h5、h6。
- p：HTML 段落标签。
- div：HTML 层标签。
- strong：定义要强调显示的内容。

4) HTML 列表标签

- ul：定义 HTML 列表。
- li：定义 HTML 列表内容。

5) HTML 表格类标签

- table：定义 HTML 表格。
- tr：定义表格行。
- td：定义表格列。

6) HTML 表单类标签

- form：HTML 表单标签。
- input：定义一个表单的输入域。
- select：定义可选择的 HTML 表单。
- textarea：定义一个多行的文字输入域。

以上 20 个标签，如果能全部掌握，网页中 95%的 HTML 代码就掌握了。如果遇到生疏的 HTML 代码，可以自己去 Google。

HTML 标签主要分为两类：两边封闭的(如 html)和自封闭的。自封闭的 HTML 标签非

成对出现(单独出现)，而且以 “/>” 结尾，称为自封闭。所有自封闭的 HTML 标签如下：br、hr、col、img、area、base、link、meta、frame、input、param、DOCTYPE、isindex、basefont。

3. HTML 属性

HTML 属性一般都出现在 HTML 标签中，HTML 属性是 HTML 标签的一部分。

标签可以有属性，它包含了额外的信息。属性的值一定要在双引号中。标签可以拥有多个属性，属性由属性名和值成对出现。

HTML 属性的语法：

```
<标签名 属性名 1="属性值" 属性名 2="属性值" ... 属性名 N="属性值"></标签名>
```

例如：

```
    <a href="http://howwe.net/">howwe.net</a>
```

标签<a>是超链接标签，使用 href 属性可以定义链接的位置(URI)。

4. HTML 元素

HTML 元素是构建网页的一种单位，由 HTML 标签和 HTML 属性组成，HTML 元素也是网页中的一种基本单位。

例如：

```
<a href="http://howwe.net">人生需要引导</a>
```

这是一个 HTML 链接元素。

```
<p>这是我的第一个网页，来吧
    <a href="http://test.howwe.net">尽情学习</a>吧!
</p>
```

这是一个 HTML 段落元素，它包含了一个 HTML 链接元素。

5. HTML 文档

HTML 文档就是 HTML 页面，也就是网页，由 HTML 元素组成。互联网上的所有内容都是由一个个的 HTML 文档呈现的。

6. HTML 注释

我们经常要在一些代码旁做一些 HTML 注释。这样做的好处有很多：方便查找比对以及项目组的其他程序员了解你的代码，而且可以方便你以后对自己代码的理解与修改等。

HTML 注释使用<!--开始，使用-->结束，语法如下：

```
<!--注释的内容-->
```

9.3　JavaScript 语法

1. JavaScript 简介

JavaScript 简称 JS，是一种基于对象和事件驱动的客户端脚本语言，起源于 Netscape 公司的 LiveScript 语言，最初是为了检验 HTML 表单输入的正确性。

如图 9-1 所示，JavaScript 由 ECMAScript(语法)、Browser Objects(DOM、BOM)组成，可以检测表单的正确性，实现 Ajax，读、写、改变 HTML 页面的架构 DOM，对事件作出响应，检测浏览者所使用的设备，产生很酷很炫的网页效果(DHTML)等。

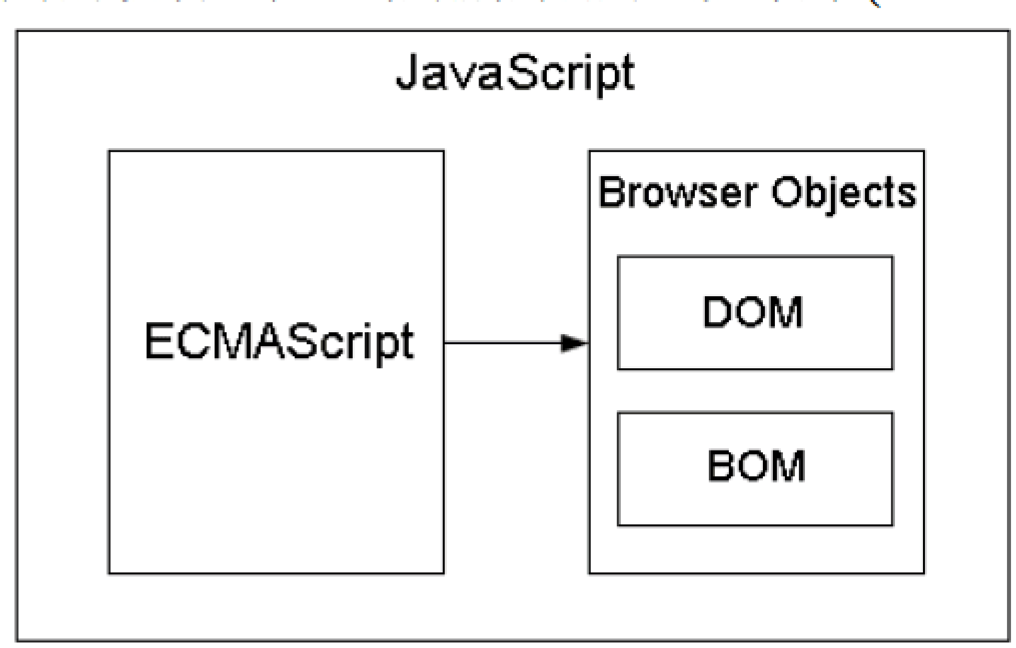


图 9-1　JS 的组成

JavaScript 与 Java 没有任何关系，它们本身就是两种不同的语言，之所以用差不多的名字，主要是因为商业上的原因。JavaScript 是一种客户端脚本语言，而 Java 是服务器端语言。

2. JavaScript 相关知识点

以下是 JavaScript 相关知识点的列表，从中可以看出，JavaScript 也包含数据类型、控制结构、数据结构三部分。

1　在 HTML 中嵌入 JavaScript
  1.1　内部引用 JavaScript
  1.2　外部引用 JavaScript
  1.3　内联引用 JavaScript
2　JavaScript 语法
  2.1　JavaScript 注释
  2.2　JavaScript 变量
  2.3　JavaScript 数值类型

2.4 JavaScript 字符串类型
2.5 JavaScript 运算符与表达式
2.5.1 JavaScript 算术运算符与表达式
2.5.2 JavaScript 赋值运算符与表达式
2.5.3 JavaScript 自增、自减运算符与表达式
2.5.4 JavaScript 逗号运算符与表达式
2.6 JavaScript 程序设计
2.7 JavaScript 顺序程序设计
2.8 JavaScript 选择程序设计
2.8.1 JavaScript 布尔类型
2.8.2 JavaScript 关系运算符与表达式
2.8.3 JavaScript 逻辑运算符与表达式
2.8.4 JavaScript if...else 语句
2.8.5 JavaScript 条件运算符与表达式
2.8.6 JavaScript switch...case 语句
2.9 JavaScript 循环程序设计
2.9.1 JavaScript while 语句
2.9.2 JavaScript do...while 语句
2.9.3 JavaScript for 语句
2.9.4 JavaScript break 与 continue 语句
3 JavaScript 面向对象(OO)语法
3.1 JavaScript 面向对象代码实践
3.2 使用构造函数创建 JavaScript 对象
3.3 JSON 法创建 JavaScript 对象

3. JavaScript 的嵌入方式

1) 内部引用：通过 HTML 的 script 标签加载 JavaScript 代码

例 1002.htm：

```
<head>
   <script type="text/javascript">
      document.write("welcome!");
      alert("welcome!");
   </script>
</head>
```

可以通过注释隐藏 JavaScript 代码。

上面范例可改为 1003.htm：

```
<head>
    <script type="text/javascript">
    <!--
        document.write("welcome!");
        alert("welcome!");
    //-->
    </script>
</head>
```

`<!-- ... //-->`用于当浏览器不支持 JavaScript 时，屏蔽 JavaScript 代码。

现在，这种隐藏 JavaScript 代码的方式可以忽略，因为所有浏览器都支持 JavaScript，除非用户手动禁止浏览器的 JavaScript 功能，否则这种情况很少会发生。

❖ **注意：**

在 XHTML 文档中，可以使用 CDATA 来避免解析 HTML 实体，即使用 CDATA 囊括的内容中的 HTML 实体将不被翻译。具体内容请自己 Google。

高级用法：使用 noscript 标签为用户提供更好的体验

通过 JavaScript 注释的方式可以隐藏 JavaScript 代码，通过 noscript 标签可以为用户提供更好的体验(提示你的浏览器不支持 JavaScript)。

例 1004.htm：

```
<body>
    <script type="text/javascript">
        document.write("welcome!");
    </script>
    <noscript>
        <p>浏览器的 JavaScript 功能已被禁用或不支持 JavaScript，请更改浏览器选项以
        启用 JavaScript。
        </p>
    </noscript>
</body>
```

2) 外部引用：引用 HTML 文件外部的 JavaScript 文件

这种方式可以使代码更清晰，更容易扩展。通过 HTML 的 script 标签加载 JavaScript 文件。

标准方法是把 JavaScript 文件放到 head 标签内，JavaScript 文件必须使用 js 作为文件名后缀。

```
<head>
    <script type="text/javascript" src="hwui.js"></script>
</head>
```

为防止网页加载缓慢，也可以把非关键的 JavaScript 放到网页底部。

外部引用的优点：统一定义 JavaScript 代码，方便查看和维护；使代码更安全，可以压缩，加密单个 JavaScript 文件；浏览器可以缓存 JavaScript 文件，减少宽带使用(当多个页面同时使用一个 JavaScript 文件的时候，通常只需下载一次)。

❖ 注意：

尽量不要把 JavaScript 分为多个文件，多个文件会增加服务器的负担，增加服务器的 HTTP 请求。

3) 内联引用：通过 HTML 标签中的事件属性来实现

例 1005.htm：

```
<input type="button" value="测试" onclick="alert('你单击了一个按钮');">
```

上面示例将调用 input 标签的 onclick 属性，弹出一个提示框。

4. 常用的 JavaScript 语句

请参见附录。

9.4　CSS 语法

CSS 是 Cascading Style Sheets 的缩写，中文名为层叠样式表。CSS 是一种标记语言，不需要编译，可以直接由浏览器执行(属于浏览器解释型语言)。

CSS 用于网页布局及网页美化。可以通过简单更改 CSS 文件，来改变网页的整体表现形式。这样可以减少我们的工作量，所以 CSS 是每一个网页设计人员的必修课。

CSS 文件是文本文件，它包含 CSS 标记。CSS 文件必须使用 css 作为文件名后缀，对大小写不敏感，由 W3C 的 CSS 工作组产生和维护。

CSS 语法非常简单，组成 CSS 语法的元素只有 CSS 选择符和 CSS 属性。每个 CSS 选择符由一个或多个 CSS 属性组成。

以下是 CSS 相关知识点的列表：

1　CSS 语法——最基本的知识

　1.1　外部引用 CSS

　1.2　内部引用 CSS

　1.3　内联引用 CSS

　1.4　CSS 选择符

　1.5　CSS 声明

　1.6　CSS 注释

2 CSS color 属性——CSS 前景色

CSS opacity 属性

3 CSS 颜色—— 五颜六色的世界

3.1 CSS RGB(A)颜色

3.2 CSS HSL(A)颜色

3.3 短 16 进制颜色与 Web 安全色

4 CSS 背景

4.1 CSS background-color 属性

4.2 CSS background-image 属性

4.3 CSS background-repeat 属性

4.4 CSS background-position 属性

4.5 CSS background-attachment 属性

4.6 CSS background 属性

5 CSS 文本

5.1 CSS letter-spacing 属性

5.2 CSS word-spacing 属性

5.3 CSS text-decoration 属性

5.4 CSS text-transform 属性

5.5 CSS text-align 属性

5.6 CSS text-indent 属性

5.7 CSS white-space 属性

6 CSS 字体

6.1 CSS font-family 属性

6.1.1 CSS family-name 系列性字体名称

6.1.2 CSS generic-family 一般性字体名称

6.2 CSS font-size 属性

6.3 CSS font-style 属性

6.4 CSS font-variant 属性

6.5 CSS font-weight 属性

6.6 CSS font 属性

7 CSS 列表

7.1 CSS list-style-type 属性

7.2 CSS list-style-image 属性

7.3 CSS list-style-position 属性

7.4 CSS list-style 属性

8 CSS cursor 属性—— 鼠标样式

9 CSS 边框
  9.1 CSS border-width 属性
  9.2 CSS border-color 属性
  9.3 CSS border-style 属性
  9.4 CSS border 属性
10 CSS margin 属性——CSS 边外补白
11 CSS padding 属性——CSS 边内补白

下面简单介绍下 CSS 相关的知识点：

1) 使用 link 标签引用 CSS

例如：

```
<head>
   <link rel="stylesheet" type="text/css"
         href="http://www.howwe.net/style.css" />
</head>
```

2) 内部引用 CSS，使用 style 标签直接把 CSS 文件中的内容加载到 HTML 文档内部

例如：

```
<style type="text/css"><!--
/* ----------文字样式开始---------- */

/* 说明一 */
.white12px
{
   color:white;
   font-size:12px;
}
/* 说明二 */
.black16px
{
   color:black;
   font-size:16px;
}

/* ----------文字样式结束---------- */
-->
</style>
```

❖ **注意：**

style 标签应该在 head 标签内部。

3) 内联引用可以把 CSS 样式直接作用在 HTML 标签中

例如：

```
<p style="font-size: 10px; color: #FFFFFF;">
使用 CSS 内联引用表现段落。
</p>
```

4) CSS 选择符——CSS 的名称

CSS 选择符就是 CSS 样式的名称，当在 HTML 文档中表现一个 CSS 样式的时候，就要用到一个 CSS。怎么用呢？就是通过 CSS 选择符(CSS 的名称)指定这个 HTML 标签使用此 CSS 样式。

选择符语法：

```
选择符名称
{
    声明;
}
```

5) 选择符的命名规则

CSS 选择符必须以字母开头，使用英文字母的大写或小写、数字、连字符、下划线、冒号、句号等组合而成。

常用的 CSS 选择符有三种：

- xhtml 标签选择符，比如 p 标签选择符(表示所有的段落都使用这个 CSS 样式)，如 `p{font-size:12px;}`。
- id 选择符，也称唯一性选择符。如 `#red{color:red;}`，在名称前增加了一个#。id 选择符在一个页面中只能出现一次，在整个网站中也最好只出现一次(这样有利于程序员控制此元素，如果有多个名称一样的元素，就无法分开、不好控制了)。
- class 选择符，多重选择符。如`.blue{color:blue;}`，在名称前增加了一个点号(.)，class 选择符在一个页面中可以出现多次。

表 9-1 对 CSS 选择符的用法做了总结。

表 9-1　选择符用法总结

选择符类型	CSS 中的写法	XHTML 中的写法
xhtml 标签选择符	p{font-size:12px;}	www.howwe.net
id 选择符	#id_selector{font-size:12px;}	<p id="id_selector">梦之都</p>
class 选择符	.class_selector{font-size:12px;}	<p class="class_selector">梦之都</p>

6) CSS 声明是由属性、冒号(:)、属性值和分号(;)组成的

语法：

```
属性:属性值;
```

例如：

```
font-size:12px;
```

font-size 表示字体大小，12px 表示字体大小的值。

7) 两个常用的技巧

- 选择符的名称一样，声明可以组合。
- 声明全部一样，选择符的名称可以组合。

9.5 快速掌握 PHP

我们先看看下面的例 1007.php：

```
<?php
require once 'inc/common.php';

$artdb = array();
$sql = "select id,title from hw articles order by id desc";

$result = $db->query($sql);
while($article = $db->fetch_array($result)){
        $article['artid'] = $article['id'];
        $artdb[] = $article;
}
?>

<table>
<tr><td>title-文章标题</td></tr>
<?php
foreach($artdb as $art){
?>
<tr><td><a href="<?= $art['catedir']?>/<?= $art['artid']?>.html"><?=
        $art['title']?></a>
    </td>
</tr>
<?php
```

```
    }
    ?>
    </table>
```

执行后的页面内容截图如图 9-2 所示。

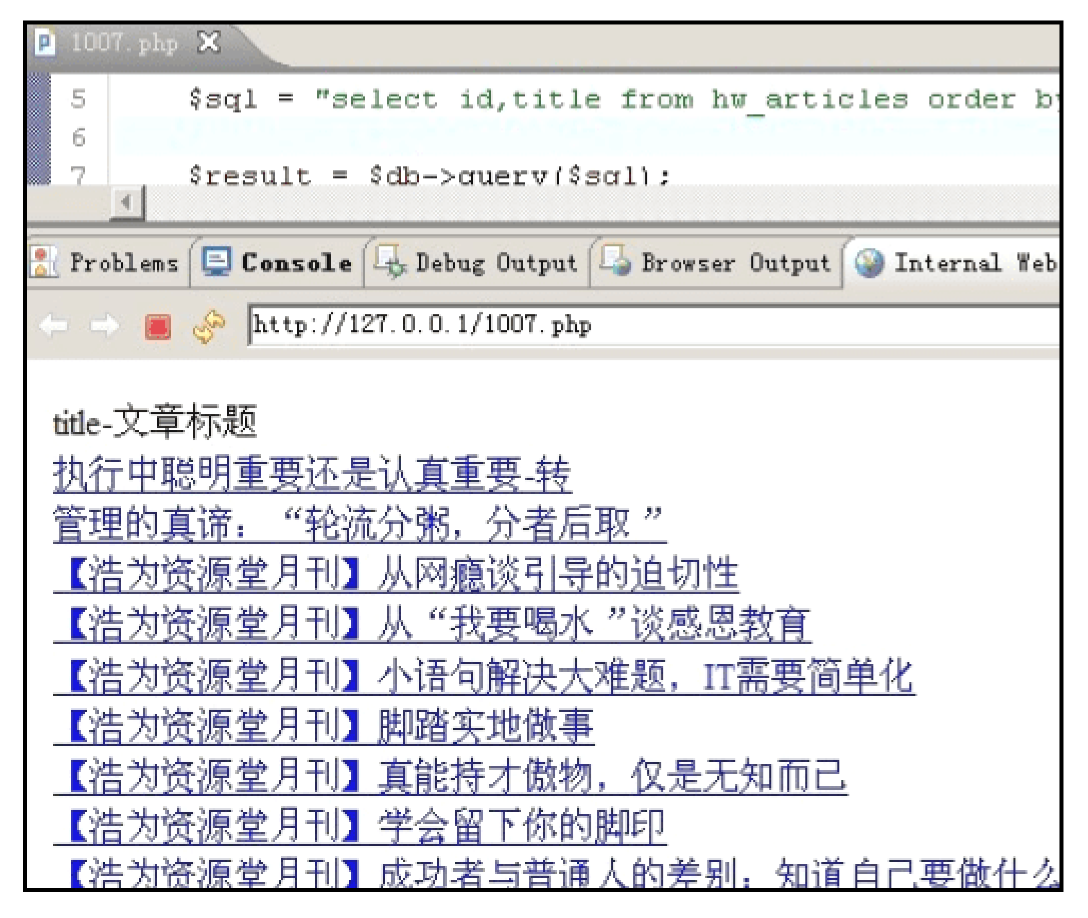


图 9-2　执行后的页面截图

从上面的代码可粗略看出一个 PHP 页面可以分为以下几部分：

- 静态数据，如 <tr><td>title-文章标题</td></tr>。
- PHP 指令，如 require_once。
- PHP 变量，如$artdb。
- PHP 动作。

静态数据

静态数据就是不变的数据，文件中的内容和输出给 HTTP 响应的内容完全一致，客户端每次请求都会得到相同的响应内容。

PHP 指令

PHP 指令控制编译器如何生成脚本文件，参见“第 5 章 控制结构”中阐述的指令，此处不作说明。

脚本元素和变量

标准脚本变量，以下是永远可用的脚本变量：

- Request：HTTP request 对象。
- response：HTTP response 对象。
- session：HTTP session 对象，在多个请求间追踪某个客户的信息。

PHP 动作

PHP 动作可以简单地看成如何处理数据，如例 1007.php 中对文章标题的显示。

9.6 PHP 常见对象

在进行 PHP 页面的处理时，一般要注意几大对象，其中每个对象都有一个使用范围。使用范围定义了在什么时间内，在哪一个页面可以访问这些对象。例如，session 对象在会话期间，可以在多个页面中被访问。

本节简单说明以下 5 大对象：$_GET、$_POST、$_REQUEST、cookie 和 session。

1. $_GET 变量

用于收集来自 method="get"的表单中的值，$_GET 变量是一个数组，内容是由 GET 方法发送的变量名和值。从带有 GET 方法的表单发送来的信息，对任何人来说都是可见的(会显示在浏览器的地址栏中)，但对发送的信息量有限制(最多 100 个字符)。

例 1008.htm：

```
<form action="1009.php" method="get">
Name: <input type="text" name="name" />
Times: <input type="text" name="times" />
<input type="submit" />
</form>
```

当用户单击“提交”按钮时，发送的 URL 如下：

```
http://127.0.0.1/1009.php?name=howwe&times=2010
```

1009.php 文件现在可以通过$_GET 变量获取表单数据了(请注意，表单域的名称会自动成为 $_GET 数组中的 ID 键)。

例 1009.php：

```
Welcome <?php echo $ GET["name"]; ?>.<br />
You have requested <?php echo $_GET["times"]; ?> times!
```

在使用$_GET 变量时，所有的变量名和值都会显示在 URL 中。所以在发送密码或其他敏感信息时，不应该使用这种方法。不过，正因为变量显示在 URL 中，因此可以在收藏夹中收藏该页面。

❖ **注意：**

HTTP GET 方法不适合大型的变量值，值不能超过 100 个字符。

2. $_POST 变量

用于收集来自 method="post"的表单中的值，$_POST 变量是一个数组，内容是由

HTTP POST 方法发送的变量名和值。从带有 POST 方法的表单发送来的信息，对任何人来说都是不可见的(不会显示在浏览器的地址栏中)，并且对发送的信息量也没有限制。

例 1010.htm：

```
<form action="1011.php" method="post">
Name: <input type="text" name="name" />
Times: <input type="text" name="times" />
<input type="submit" />
</form>
```

当用户单击“提交”按钮时，发送的 URL 如下：

```
http://127.0.0.1/1011.php
```

1011.php 文件可以通过$_POST 变量获取表单数据了(请注意，表单域的名称会自动成为 $_POST 数组中的 ID 键)

例 1011.php：

```
Welcome <?php echo $_POST["name"]; ?>.<br />
You have requested <?php echo $_POST["times"]; ?> times!
```

为什么使用 $_POST？原因有二：

1) 通过 HTTP POST 发送的变量不会显示在 URL 中。

2) 变量没有长度限制。

不过，由于变量不显示在 URL 中，所有无法把页面加入收藏夹。

3. $_REQUEST 变量

$_REQUEST 变量包含了$_GET、$_POST 以及$_COOKIE 等，可用于取得通过 GET 或 POST 方法发送的表单数据的结果。

例 1012.php：

```
Welcome <?php echo $_REQUEST["name"]; ?>.<br />
You have requested <?php echo $_REQUEST["times"]; ?> times!
```

4. cookie

cookie 常用于识别用户。cookie 是服务器留在用户计算机中的小文件。每当相同的计算机通过浏览器请求页面时，它同时会发送 cookie。通过 PHP，可创建并取回 cookie 的值。

1) 如何创建 cookie？

setcookie()函数用于设置 cookie。

❖ **注意：**

setcookie()函数必须位于<html>标签之前。

语法：

```
setcookie(name, value, expire, path, domain);
```

例 1013.php(创建名为 user 的 cookie，并赋值 Howwe，规定一小时后过期)：

```
<?php
setcookie("user", "Howwe", time()+3600);
?>

<html>
<body>
cookie 测试
</body>
</html>
```

❖ **注意：**

发送 cookie 时，cookie 的值会自动进行 URL 编码，而在取回时进行自动解码(为防止 URL 编码，可使用 setrawcookie())。

2) 如何取回 cookie 的值？

$_COOKIE 变量用于取回 cookie 的值。

例 1014.php(取回名为 user 的 cookie 的值，并把它显示在页面上)：

```
<?php
// Print a cookie
echo $_COOKIE["user"]."\n<br>";

// A way to view all cookies
print_r($_COOKIE);
?>
```

例 1015.php(可首先使用 isset()函数确认是否设置了 cookie)：

```
<html>
<body>

<?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br />";
```

```
else
  echo "Welcome guest!<br />";
?>

</body>
</html>
```

3) 如何删除 cookie？

使过期日期变更为过去的时间点即可。

例 1016.php(删除 cookie)：

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
echo "del cookie ok!<br>";
?>
```

4) 如果浏览器不支持 cookie 怎么办？

如果浏览器不支持 cookie，就得采取其他方法在应用中从一个页面向另一个页面传递信息了。有一种方式是从表单传递数据。可参考例 1008.htm 和 1009.php。

5. session

session 变量用于存储有关用户会话的信息或更改用户会话的设置。session 变量保存的信息针对单一用户，并可供应用程序中的所有页面使用。

当运行一般应用程序时，打开后，作些更改，然后关闭。这一过程可简称为一次会话：计算机清楚操作者是谁，也知道是何时启动应用程序，以及何时终止。但在互联网上，由于 HTTP 地址不能维持状态，所以服务器不知道操作者是谁以及在做什么。

为了解决这个问题，PHP 通过 session 在服务器上存储用户信息(如用户名称、购买商品等)，以便相应页面能使用。但 session 信息是临时的，用户离开网站后将被删除。如果需要永久存储，可将数据存储在数据库中。

session 工作原理：为每个访问者创建一个唯一的 id (UID)，并基于这个 UID 来存储变量。UID 存储在 cookie 中，可通过 URL 进行传导。

1) 开始 session

在将用户信息存储到 session 中之前，必须启动会话。session_start()函数必须位于<html>标签之前。

2) 存储 session 变量

存储和取回 session 可用$_SESSION 变量。

例 1017.php(启动会话并存储 session)：

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>

<html>
<body>

<?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>

</body>
</html>
```

输出内容：

```
Pageviews=1
```

在下面的例子中，可以创建一个简单的 page-view 计数器。isset()函数用来检测是否设置了 views 变量。如果已设置，则累加计数器。如果不存在，则创建 views 变量，并设置为 1。

例 1018.php(page-view 计数器)：

```
<?php
session_start();

if(isset($_SESSION['views']))
  $_SESSION['views']=$_SESSION['views']+1;

else
  $_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

3) 删除 session

如果要删除某些 session 数据，可以使用 unset()或 session_destroy()函数。

unset()函数用于释放指定的 session 变量：

```
<?php
unset($_SESSION['views']);
?>
```

也可以通过 session_destroy()函数彻底删除 session：

```
<?php
session_destroy();
?>
```

❖ **注意：**

session_destroy()将重置 session，所有已存储的 session 数据都将无效。

9.7　学什么：学以致用

现在许许多多的初学者和程序员，都在趋之若鹜地学习 Web 开发的宝典级框架——SSH，似乎这个框架成了一个人是否精通 Java，能否写 J2EE 程序的唯一事实标准和找工作的必备基础。不少人对 PHP 的学习，沉迷于对框架的了解，却将基本的 PHP 语法拒之门外。

然而，如果在面试的时候问这些程序员，你们为什么要学习这些框架？这些框架的本质到底是什么？似乎很少有人能够给出满意的答复。因为他们都在为了学习而学习，为了工作而学习，而没用真正去了解一个框架。

其实所有的人都应该思考这样的问题：为什么要学习框架？框架到底给我带来了什么？

无论是使用 JSP，还是使用 Struts、SSH，我们至少都需要一些必需的元素，如果没有这些元素，或许我们真不知道程序能写成什么样子。

1. 数据

在 Hello World 示例中就是 Cnt，它们共同构成了程序的核心载体。事实上，我们往往会用一个 Hello 类封装 Cnt，这使得我们的程序更加 OO(Object-Oriented，面向对象)化。不管怎么说，数据都会穿插在这个程序的各个地方，成为程序运行的核心。

2. 页面展示

在页面上，我们需要利用 HTML 把需要展现的数据都呈现出来。同时我们还需要完成一定的页面逻辑，例如错误展示、分支判断等。

3. 处理具体业务的场所

不同的框架，处理具体业务的场所不一样。如 Java 中原来用 JSP 和 Servlet，后来用 Struts 的 Action。PHP 中原来直接用 PHP 代码的，转而用框架的某个功能。

上面所讲的这些必须出现的元素，在不同的年代被赋予了不同的表现形式，有的受到时代的束缚，其表现形式非常落后，有的已经不再使用。但是拨开这些外在的表现形式，我们就可以发现，这不就是 MVC 吗？

数据	->	**Model**
页面展示	->	**View**
处理具体业务的场所	->	**Control**

所以，框架不重要，概念才是王道。只要能够深刻理解 MVC 的概念，框架对你来说，只是一堆 jar 包而已。

MVC 模式的概念很简单，并且这些概念早已深入我们的内心，而我们所缺乏的是将其本质挖掘出来。我们来看看图 9-3 所示的模型图，这是一幅流行了很多年的讲述 MVC 模型的图。

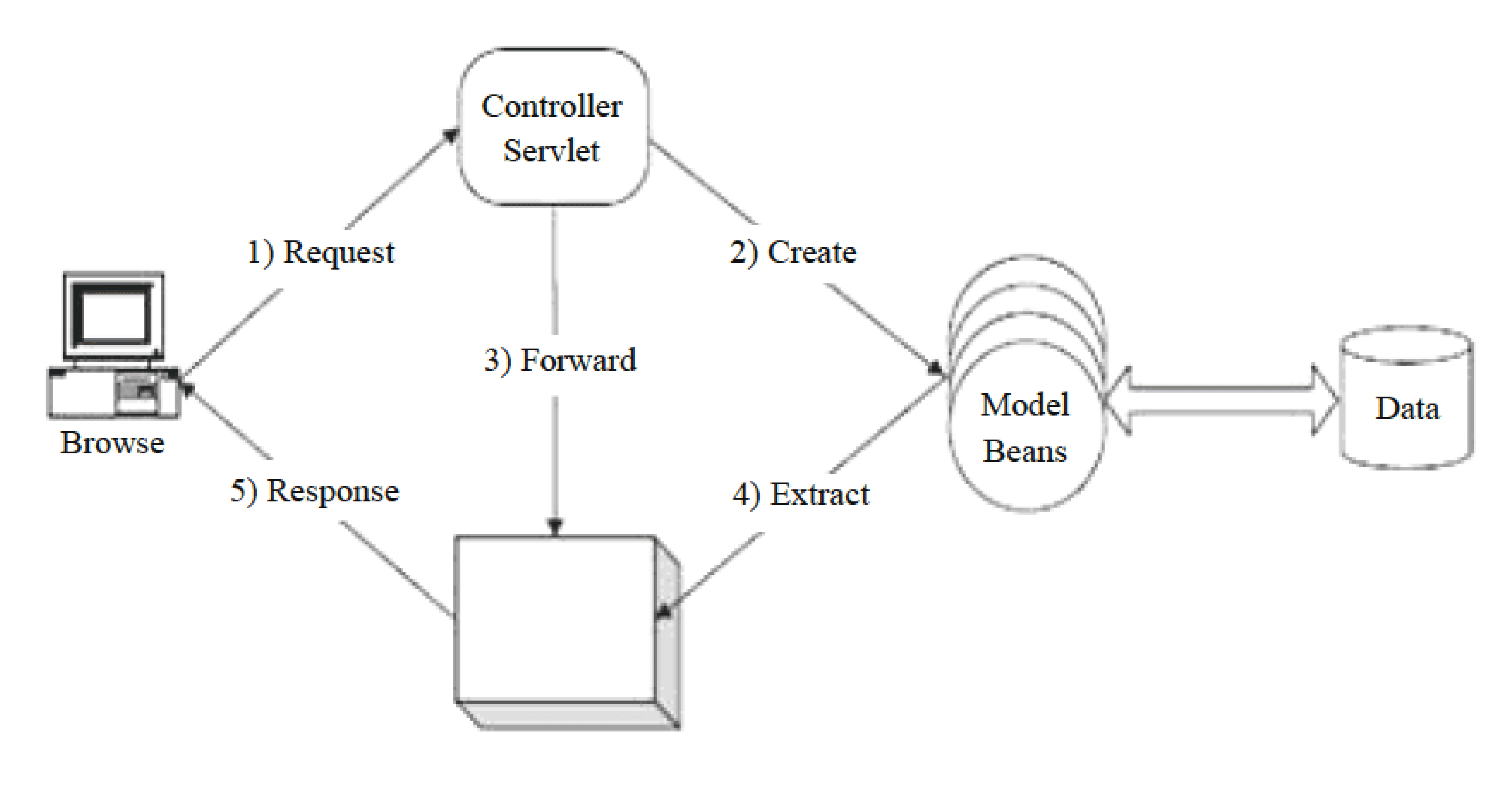


图 9-3　MVC 模型图

在这幅图中，MVC 的三个部分各司其职，结构清晰明朗。但这幅图忽略了一个问题，数据是动的，数据在 View 和 Control 层一旦动起来，就会产生许多的问题：

1) 数据从 View 层传递到 Control 层，如何使一个个扁平的字符串转换成一个个生龙活虎的 Java 对象？

2) 数据从 View 层传递到 Control 层，如何方便地对数据格式和内容进行校验？

3) 数据从 Control 层传递到 View 层，一个个生龙活虎的 Java 对象，又如何在页面上以各种各样的形式展现出来？

4) 如果试图将数据请求从 View 层发送到 Control 层，如何才能知道你要调用的究竟是哪个类、哪个方法？一个 HTTP 请求，又如何与 Control 层的 Java 代码建立起关系？

除此之外，Control 层似乎也没有想象中的那么简单，因为它作为一个控制器，至少还需要处理以下的问题：

1) 作为调用逻辑处理程序的控制者，如果逻辑处理程序发生了异常，我们该如何处理？

2) 对于逻辑处理的结果，我们需要做什么样的处理才能满足丰富的前台展示需要？

这一个又一个问题的提出，都基于对 MVC 基本概念的挖掘。所以，这些问题都需要我们在写程序的时候去一一解决。

说到这里，前面所提的问题应该有了答案：框架是为了解决一个又一个在 Web 开发中所遇到的问题而诞生的。不同的框架，都是为了解决不同的问题，但是对于程序员而言，它们只是 jar 包而已。框架优缺点的评论，也完全取决于其对问题解决程度和解决方式的优雅性的评论。所以，千万不要为了学习框架而学习框架，而是要为了解决问题而学习框架，这才是一个程序员的正确学习之道。

拿设计模式来说，我就没去看过几个设计模式，很多设计模式对我没什么用处，我也用不着去学习。只有在解决问题时，需要什么技术，我才会去学。学以致用，学是拿来用的，不是用来消磨时间的。虽然无数的人都说学习 Java 编程必学 GOF 设计模式，但是我认为对我来说没用，我就没有去看，就是到现在我也说不出几种设计模式。

J2EE 设计模式能解决不少问题，所以我花了不少时间去学习，但也没有完全照搬整个模式，只是借鉴了其设计思想，尽量简单化，形成自己的系统构架 HoCAS。

对于企业来说，买一个软件产品，他不关心构架，只关心软件的成本及使用是否方便。推销软件的厂商在介绍时，大多数会说应用了什么技术、使用了何种架构，如何如何先进。可是对于公司的决策者来说，那些只会听得他昏昏欲睡，他关心的是成本！购买了该软件能给公司带来多大效益，投资需要多少，为了满足需要我们还得购买什么。

动辄就是多少个 CPU，多少内存，难道真的需要吗？作为用户，我不关心软件是否容易修改，我要的是高效的实现、快速的反应、很低的故障率、易维护而健壮的程序。所以请程序员们清醒起来，不要因为架构而沾沾自喜，不要因为技术领先而自我欣赏，对用户而言没有任何的意义。用户很少在架构上进行二次开发，即使开发也不可能真的理解了架构后再去，他们只需要一个接口就 OK 了。

第 10 章

Web RIA 简介

Web 表现层变化频繁，单纯 PHP 或 J2EE 框架已无法满足需求，并且严重影响了开发及使用的效率。

RIA(Rich Internet Applications，富互联网应用)技术，结合桌面应用程序反应快、交互性强的优点与 Web 应用程序的传播范围广及容易传播的特性，完全解决了 J2EE 等框架在表现层的劣势。RIA 充分利用 Ajax 技术，将 Web 表现层完全或者绝大部分前推到客户端来开发，大大简化服务器端的 PHP 或 Java 开发。

RIA 有三大主流技术：

- Adobe 的 Flash/Flex，统称为 AIR。
- 微软的 Silverlight，WPF 的一部分。
- OpenAjax 联盟。

另外还有 Sun 的 JavaFX，由于已经不被很多人看好，此处不介绍。

RIA 的优势

RIA 具有桌面应用程序的特点：在消息确认和格式编排方面提供互动的用户界面；在无刷新页面之下提供快捷的界面响应时间；提供通用的用户界面特性(如拖放式(drag and drop))以及在线和离线操作能力。

RIA 同时还具有 Web 应用程序的特点：立即部署、跨平台、采用逐步下载来检索内容和数据，以及可以充分利用被广泛采纳的互联网标准。RIA 具有的通信特点，则包括实时互动的声音和图像。

客户机在 RIA 中的作用不仅是展示页面，它还可以在幕后与用户请求异步地进行计算、传送和检索数据、显示集成的用户界面和综合使用声音及图像，这一切都可以在不依靠客户机连接的服务器或后端进行。

对于企业来说，部署 RIA 的好处在于：

1) RIA 可以继续使用现有的应用程序模型(包括 J2EE 和.NET)，因而无需大规模替换

现有的 Web 应用程序。通过 Rich Client 技术，可以轻松构建更为直观、易于使用、反应更迅速并且可以脱机使用的应用程序。

2) RIA 可以帮助企业提供多元化的重要业务效益，包括提高产销量、提高品牌忠诚度、延长网站逗留时间、实现较频繁的重复访问、减少带宽成本、减少支持求助以及增强客户关系等。

RIA 单从技术层面来说比较乏味，但使用 RIA 的理念能简化页面的开发。比如在 HoCAS 中，页面只需要定义一个数据模型，而不用考虑具体的页面实现，代码请参考“11.5 jQuery 实战：HoCAS 表示层原理”。

10.1　RIA 三大主流技术

1. Flash/Flex(AIR)

Flash/Flex 早先是由 Macromedia 开发出来的。但 Adobe 目光独到，2005 年把 Macromedia 收购了，于是 Flash/Flex 就成了 Adobe 的如意法宝。其实，当初收购 Macromedia 的价格并不高，也就 34 亿美金。Adobe 买下 Macromedia 后，对 Flash 下了大本钱，再加上那几年没有直接的竞争对手，所以到 2007 年，Flash/Flex 已经成为 RIA 市场的事实标准，当时的 PC 占有率少说也有 70%，而且已经开始进入手机市场。

目前 Flash/Flex 的主要优势有两个：一个是用户群大，另一个是跨平台(包括操作系统、浏览器、移动设备)。

但不少人用过后，感觉 Flash/Flex 也存在不少问题。一个主要的问题是语言的兼容性不够好，从 ActionScript 2 迁移到 ActionScript 3，很多代码几乎都要重写；还有一个问题是功能不够强，让人感觉很不爽。比如，至今仍不能够很好地支持多线程(仅支持异步回调)，不能很好地整合 PDF(照理说都是自家公司产品，整合应该不难)。

2. Silverlight(WPF)

到了 2006 年后，微软发觉苗头不对，赶紧下大力气自己搞。在 2007 年底和 2008 年底，分别发布了 Silverlight 1.0 和 Silverlight 2.0。然后商务层面也接连出手：先是 2008 年奥运期间与 NBC(美国国家广播公司)合作，用 Silverlight 进行赛事直播；接着在奥巴马就职典礼上，也用了 Silverlight。

微软的意图非常明显，那就是：在市场方面利用各种机会争夺用户占有率，弥补对 Flash 的劣势；在技术方面不断强化功能，力图甩开 Flex，吸引开发人员加入。

Silverlight 的主要优势是依托.NET。借着.NET 这个靠山，Silverlight 能整合现有的某些语言(据说已能支持 JavaScript、IronPython、IronRuby、VB)和库，还能够方便原有的.NET 程序员上手；Silverlight 在功能上也显得比 Flex 更强大(比如多线程和 3D 方面)。

不过，依托于.NET 也导致了 Silverlight 的主要缺点：跨平台性很差。虽说现在有 Moonlight 的帮忙，但依然不够理想(尤其是对 Linux 的支持)。

3. OpenAjax 联盟

呼声很高、也被很多人看好的是 OpenAjax 联盟(http://www.openajax.org/)，这是一个专注于 XML 和 JavaScript 技术开发的团体。目前 OpenAjax 联盟由 Adobe、BEA、IBM、Mozilla、Novell、Opera、Oracle、SAP、Sun Microsystems、Tibco、Zend 等 30 多位主流软件厂商组成。

因为 OpenAjax 更能体现 RIA 的优势，可以继续使用现有的应用程序模型(包括 J2EE 和.NET)，因而无需大规模替换现有的 Web 应用程序。

可以说正是由于 Ajax 应用的兴起，RIA 才被拉入广大用户的视线。OpenAjax 能够与 HTML 进行无缝集成，并可用于支持 HTTP 的任何应用程序平台。

如果采用 RIA 开发网站，则与传统的 Web 开发会有两点很大的区别：

- 对于服务端，View 层的依赖没有了。
- 不需要在 Server 端保存 Session 信息。

无论是 Rails，还是 PHP、Python、Java 的各种 Web 框架，Session 和 Server View 都是必需的。从上个世纪末期到现在 10 多年的时间里，Web 的发展和 HTTP 本身无状态的特点，使得我们无法脱离这两点来开发网站。几乎所有的 Web 框架都会涉及它们，但是都会采用各自的方法来解决。每个方案都没有本质上的不同，却在形式上差别很大，Server View 技术可能是每个框架中最麻烦的部分。

RIA 却在试图打破这个局面。一旦采用了 RIA，就会发现，各种框架其实在编写业务逻辑和服务方面(ORM 层，Router)并没有什么显著的不同。单从 Controller 和 Model 方面考虑，脚本语言及 Web 框架还是有一定敏捷优势的。

在过去的两到三年中，Web 开发人员一直想构建一种比传统 HTML 更丰富的客户端：这是一个用户接口，它比用 HTML 实现的接口更加健壮、反应更加灵敏，并且具有更令人感兴趣的可视化特性。

RIA 技术的出现，允许我们在 Internet 上以一种像使用 Web 一样简单的方式来部署富客户端程序。无论将来 RIA 是否能够如人们所猜测的那样完全代替 HTML 应用系统，但是，对于那些采用基于 C/S 架构的胖客户端技术运行复杂应用系统的机构，以及采用基于 B/S 架构的瘦客户端技术部署 Web 应用系统的机构来说，RIA 确实提供了一种廉价的选择。

后续章节中的“11.8　Gears、AIR、WPF 的异同”对 AIR 及 WPF 有更详细的阐述，欢迎浏览。

10.2　OpenAjax Hub

OpenAjax 联盟致力于推广 Ajax 通用的兼容性的潜力，以及很容易结合到新的或现有

的软件程序中的特性，通过提供互操作性降低采用 Ajax 的风险，保证 Ajax 解决方案坚持走开放标准路线和使用开放源码技术，保持 Web 的开放性。

OpenAjax Hub 就是这一思想的结晶，目前版本为 2.0(下载地址为 http://www.openajax.org/)。

1. OpenAjax Hub 简介

OpenAjax Hub(简称 Hub)主要用于 Web 应用开发人员需要在同一个应用中同时使用多个 Ajax 运行库的情况。它提供标准的 JavaScript，当被包含在用 Ajax 技术创建的 Web 应用中时，让多个 Ajax 工具包能够在同一个页面里一起协同工作。

Ajax 应用开发人员在开发过程中的需求往往存在着多样性，这种情况导致了如今市场上存在超过 200 个各种各样的 Ajax 产品，同时这些产品的架构和特性也存在很大的多样性。对于一些开发者而言，他们认为开发中最重要的因素是找到一种能够和后端服务器很好集成的 Ajax 工具包。

而对于其他一些开发者而言，最重要的因素则是，是否存在可用的特定客户端组件(例如，富数据网格小部件或交互式的图表小部件)。结果，Ajax 生态系统发展到现在，开发者在大部分时间里都能找到满足他们每个特定需求的 Ajax 工具包，但是也存在问题，他们往往必须在同一个 Web 应用中混合和匹配使用多个 Ajax 工具包才能满足所有的需求。

Hub 应用的一个重要场景是门户和 Mashup，在这些场景中，应用开发人员创建一个页面，页面里松散地装配预先封装好的应用组件。Hub 实际上是保证这些 Ajax 技术创建的复杂应用，能够使用由多个不同 Ajax 工具包创建的组件进行构建。

2. 主要特性：Hub 的发布/订阅管理器

Hub 的主要特性是它的发布/订阅管理器(以下简称为“pub/sub 管理器”)。pub/sub 管理器允许 Mashup 的部分功能能够广播事件到其他应用组件的订阅中。例如，假设存在一个日历小部件，这个小部件允许用户能够选取一个特定的日期。Mashup 中可能存在多个用户界面组件，这些组件都需要根据新选择的日历日期来更新它们的可视化外观。在这种情况下，日历小部件将发布一个“新日历日期”的事件，而其他可视化小部件将订阅这个事件。因此，pub/sub 管理器的通用消息的优点是：在由不同的 Ajax 工具包所构建的组件之间提供了一个主要的集成机制。

Hub 的 pub/sub 管理器提供各种各样的高级特性，例如对事件名称通配符的强大支持。不过，下面的例子中没有展示这个特性。

示例

让我们假设现在有这样一个商业智能应用，该应用使用了以下几种 Ajax 运行库：

- UTILS.js，对浏览器的 JavaScript 环境提供非常有用的扩展，例如 XMLHttpRequest API。
- CALENDAR.js，提供一个日历小部件。
- CHARTS.js，提供一个图表小部件。

● DATAGRID.js，提供一个交互式数据网格小部件。

这个应用存在唯一的日历小部件，用户可以以图表小部件的形式(例如每日状态、每周状态、每月状态和每年状态的柱状图)和数据网格小部件的形式(例如区域数据和全国数据对比，两种数据都以用户选择的感兴趣的列展示)选择其中的一些数据视图。

当用户在日历小部件中选择了一个新的日期时，用户指定的各个可视化组件(例如图表和/或数据网格小部件)都需要被更新。

实现这个应用的一种方法，是在加载其他 Ajax 库之前加载 OpenAjax Hub 的 JavaScript。例如：

```
<html>
<head>
...
<script type="text/javascript" src="OpenAjax.js"/>
<script type="text/javascript" src="UTILS.js"/>
<script type="text/javascript" src="CALENDAR.js"/>
<script type="text/javascript" src="CHARTS.js"/>
<script type="text/javascript" src="DATAGRID.js"/>
...
</head>
...
```

一些 Ajax 运行期包含了 OpenAjax Hub，将 Hub 作为它们标准发布的组成部分，在这种情况下，只要特定的 Ajax 运行期的 JavaScript 在其他兼容 OpenAjax 的运行期加载之前被加载，就无需为 OpenAjax.js 使用一个单独的<script>元素。

为了能让应用正常工作，开发者需要注册一个回调函数，当用户在日历小部件中选择一个新的日期时，该回调函数将被调用。接着这个回调函数使用 OpenAjax Hub 的 publish()函数广播新日期事件：

```
<script type="text/javascript">
...
function MyCalendarCallback(...) {
OpenAjax.hub.publish("myapp.newdate", newdate);
}
...
</script>
```

接着开发者需要开发一些代码，让所有的图表小部件和数据网格小部件都订阅这个新日期事件，并提供回调函数。各个回调函数将相应地更新特定的可视化小部件：

```
<script type="text/javascript">
...
function NewDateCallback(eventname, publisherData,
```

```
subscriberData) {
……更新特定的可视化小部件……
}
OpenAjax.hub.subscribe("myapp.newdate", NewDateCallback);
...
</script>
```

3. 未来支持 OpenAjax Hub 的工具包

OpenAjax 联盟正在和业内厂商一起合作，以达到 OpenAjax Hub 被广泛支持的目的。

某个特定的 Ajax 工具包可以像下面这样支持 OpenAjax Hub。

Ajax 工具包可以把 Hub 包含在内(这是一种最好的方式)。Hub 可以用小于 3KB 的 JavaScript 实现，所以一些 Ajax 工具包可以简单地捆绑 Hub，将它作为工具包的一个标准组件。

如果 Hub 在运行环境里可用，就使用它。其他一些 Ajax 工具包可能决定在发布版本中不包含 Hub，它们会检查 Hub 是否早已被加载了，如果已经加载，那么它们将直接使用 Hub 的服务。

第三方的开发者可以开发适配器。对于大多数工具包而言，它们可能允许第三方的开发者编写少量的 JavaScript，使得自己能够支持 Hub。

当 Ajax 工具包包含对 Hub 的内置支持时，应用开发人员的任务将变得更加容易，但是通过查找或者编写一个简单的适配器，Hub 仍然可以与那些还未实现对 Hub 支持的工具包一起使用。

10.3　jQuery，构建 Web RIA 程序的基础

随着 Web 2.0 及 Ajax 的发展，出现了一堆 JS 框架，其中常用的有 Prototype、YUI、jQuery、mootools 以及国内的 JSVM 等。在项目中应用这些 JS 框架，能使程序员从设计和编写繁杂的 JS 应用中解脱出来，将关注点转向功能需求而非实现细节上，从而提高项目的开发速度。

jQuery 是继 Prototype 之后的又一个优秀的 JS 框架，它由 John Resig 于 2006 年初创建，目前版本为 1.3。(下载地址为 http://jquery.com/)

相关地址如下：

- http://code.google.com/p/jquery-api-zh-cn/——jQuery API 中文版翻译。
- http://www.51toria.cn/jQueryAPI1.3/——根据关键字查询方法。

jQuery 具有以下特点：代码简练、语义易懂、学习快速、文档丰富、UI 操作简洁，能将 JS 代码和 HTML 代码完全分离，便于代码维护和修改。

jQuery 是一个轻量级的 JS 框架，支持 CSS1、CSS 和 CSS3，以及基本的 xPath，1.3 的 min 版本大小为 54KB。它能在网页上进行简单的文档操作、事件处理，实现特效并为 Web 页面添加 Ajax 交互，是开发人员简化 RIA 开发的最佳选择。随着基于浏览器的应用程序不断代替桌面应用程序，jQuery 的使用将会不断增长。

jQuery 是跨浏览器的，它支持的浏览器包括 IE 6.0+、FF 1.5+、Safari 2.0、Opera 9.0+。

jQuery 插件(基于 jQuery 扩展其他功能)特别丰富，除了 jQuery 本身带有的一些特效外，可以通过插件实现更多功能，如表单验证、Tab 导航、拖放效果、表格排序、DataGrid，树型菜单、图像特效以及 Ajax 上传等。

jQuery 将会改变编写 JavaScript CSS 代码的方式，降低学习使用 JS 操作网页的复杂度，提高网页 JS 开发效率。不管是 JS 初学者还是资深专家，jQuery 都将是第一选择。

第 11 章

实战 jQuery

上一章简单介绍了 jQuery 的功能，在这一章我们将介绍 jQuery 的使用。

11.1　jQuery 简单范例

我们先看看网站 http://jquery.com 上面 jQuery 范例的截图，未执行时，如图 11-1 所示，点击链接 **RUN CODE** 后效果如图 11-2 所示。

图 11-1　单击前的效果

图 11-2　单击后的效果

以下是相关的 HTML 代码：

```
<div id="jq-learnNow">
    <h2>Learn <span class="jq-jquery"><span>jQuery</span></span>
        Now!</h2>
    <p>What does jQuery code look like? Here's the quick and dirty:</p>
    <div class="jq-codeDemo jq-clearfix">
<pre><code>$("p.neat").addClass("ohmy").show("slow");</code></pre>
```

```
        <a href="http://docs.jquery.com/Tutorials" class="jq-runCode">RunCode</a>

        <p class="neat"><strong>Congratulations!</strong> You just ran a snippet
of jQuery code. Wasn't that easy? There's lots of example code throughout the
<strong><a href="http://docs.jquery.com/">documentation</a></strong> on this
site. Be sure to give all the code a test run, to see what happens.</p>
        </div>
    </div><!-- /#learnNow -->
```

以下代码为图 11-1 所示页面部分的 HTML 代码。

```
        <h2>Learn <span class="jq-jquery"><span>jQuery</span></span>
            Now!</h2>
        <p>What does jQuery code look like? Here's the quick and dirty:</p>
        <div class="jq-codeDemo jq-clearfix">
        <pre><code>$("p.neat").addClass("ohmy").show("slow");</code></pre>
        <a href="http://docs.jquery.com/Tutorials" class="jq-runCode">Run Code</a>

        <p class="neat">这部分通过 CSS 而隐藏，相关代码如下：
        p.neat {
        display: none;
        clear: both;
        margin: 1em 0;
        padding: 1em 15px;

        background: #0F67A1;
        }
```

链接事件通过以下代码来绑定：

```
    $(function(){
            $('.jq-runCode').click(function(){
                eval($(this).parent().find('code').text());
                $(this).hide();
                return false;
            });
    });
```

整合效果见 http://127.0.0.1/demo/jquery/demo.htm，页面代码见 D:\howwe\wwwroot\demo\jquery\demo.htm，请自己仔细去浏览。代码包含 jQuery、JS、CSS、Div 等知识。JS 请参考“9.3 JavaScript 语法”，CSS 请参考“9.4 CSS 语法”，Div 请自己去 Google。

11.2 jQuery 入门

下面通过例 1201.htm，简单示范下 jQuery 的使用：

```
<html>
<head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>jQuery Sample</title>
    <script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function() {
    alert("Hello jQuery!");
    $("a").click(function() {alert("Hello World!");});
});
</script>
</head>
<body>
<a>Click Here,Will Show "Hello World!"</a>
</body>
</html>
```

说明： 将 jQuery 的 JS 文件导入之后(语句<script type="text/javascript" src="jquery.min.js"></script>)，我们就可以尝试 jQuery 提供的便利了。jQuery 中的 ready 函数用来为 DOM 快速载入 JS 脚本，在页面加载完成之前执行。例 1201.htm 中是先弹出对话框，再为链接(a 标签)绑定一个 click 事件。

$("a")是一个 jQuery selector(选择器)，在本例中它选择所有的 a 元素；$为 jQuery “类”的别名；$()构建了一个 jQuery 对象；$("?")查找所有?标签体、$("a")查找所有 a 标签、$("div") 查找 div 标签。请记住这个写法。click()方法为该 jQuery 对象$("a")的一个方法，它绑定了一个 click 事件到所有被选择的元素(但本例仅为一个 a 元素)，并在事件发生时执行提供的方法。

我们不需要为每个单独的元素写一个 onclick，结构(HTML)和行为(JS)拥有一个清晰的分离，就如同我们使用 CSS 来分离结构和页面效果。

运行 http://127.0.0.1/1201.htm，先显示如图 11-3 所示的窗口，再弹出一个提示框，内容为“Hello jQuery!”。

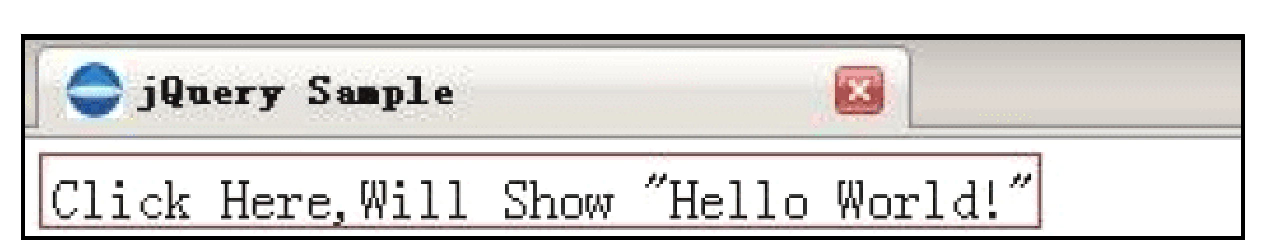


图 11-3 jQuery 范例

单击图 11-3 所示的字符“Click Here,Will Show "Hello World!"”，将显示一个提示框，内容为“Hello World!”。

对比一下常规方法是如何实现相应功能的：例 1202.htm 为最简单的写法；例 1203.htm 为标准写法。

例 1202.htm：

```
<html>
<head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>Normal Sample</title>
</head>
<body onload="alert('Hello jQuery!');">
<a href="" onclick="alert('Hello World')">Click Here,Will Show "Hello
   World!"</a>
</body>
</html>
```

❖ **说明：**

提示框直接在 body 标签的 onload 事件中实现，click 事件直接在链接(a 标签)中实现。

运行 http://127.0.0.1/1202.htm，先显示图 11-4 所示的窗口，再弹出一个提示框，内容为“Hello jQuery!”。





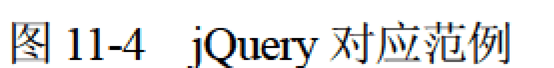


图 11-4　jQuery 对应范例　　　图 11-5　弹出的提示框

单击图 11-4 所示的链接“Click Here,Will Show "Hello World!"”，将显示一个提示框，内容为“Hello World!”，如图 11-5 所示，然后再访问页面 http://127.0.0.1/1202.htm。

例 1203.htm：

```
<html>
<head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>Normal Sample</title>
<script type="text/javascript">
alert('Hello jQuery!');
</script>
</head>
<body>
<a href="" onclick="alert('Hello World')">Click Here,Will Show "Hello
```

```
    World!"</a>
</body>
</html>
```

❖ **说明：**

提示框直接在 JS 脚本中实现，click 事件在链接(a 标签)中实现。

运行 http://127.0.0.1/1203.htm，先弹出一个提示框，内容为“Hello jQuery!”，再显示图 11-4 所示的窗口，单击链接状况类似于 1202.htm。

例 1201.htm 的优势是：JS 代码可以和 HTML 标记分开，而且很多功能使用 jQuery 很快就能实现，而用常规 JS 则需要很多代码。

小知识：jQuery 为开发带来便利

在最近某项目(其简介见 http://howwe.net/corp/150.html)的开发中，曾因觉得 jQuery 库为 50 多 KB，而另一实现 xmlhttp.js 仅为 5KB，而想将后者的实现整合进项目。于是，开始写 Java 及 JS 脚本，前后折腾了一个多小时，还没搞定；马上换 jQuery，很快就搞定。(注意：部分为 Java 代码)。

jQuery 代码如下：

```
function callajax(){
    var ch=<%=admin.getStr("TelId")%>;
    if (ch=='99')
      return;
     var vurl='<%=request.getContextPath()%>/web/db?dact=getChMb&dch=<%=
         admin.getStr("TelId")%>&duid=<%=admin.getStr("ID")%>';
     $.ajax({
         type: "POST",url: vurl,
         timeout: 30000,dataType: 'script',
         error: function(msg){},
         success: function(msg){
         }
     });
}
```

调用代码：

```
var id=window.setInterval(callajax,500);
```

Java 相应代码(主要部分)：

```
sRtn = "$(\"#shmb\").html(\""+sRtn+"\");$(\"#shkey\").html(\""+sRun+"\");";
```

❖ 注意：

shmb 和 shkey 为 span，用来显示内容。

由于 jQuery 能直接执行返回结果，所以 jQuery 的使用减少了对返回结果的解析，从而大大提高了开发效率。

11.3 jQuery 基础

本节介绍 jQuery 的基本语法，HTML 以例 1204.htm 为范本，1204.js 为相应的 JS 脚本，通过修改 1204.js 来演示 jQuery 的功能及用法。

1204.htm 的代码请参考 D:\howwe\wwwroot\1204.htm, 1204.js 的基本格式如下：

```
$(document).ready(function() {
    // do something here
});
```

运行 http://127.0.0.1/1204.htm，部分内容截图如图 11-6 所示。

图 11-6 jQuery 基础范例

1. 选择元素：selectors 和 events

jQuery 提供两种方式来选择元素：第一种，使用作为字符串传递给 jQuery 构造器的 CSS 和 XPath 选择器的联合(例如`$("div > ul a")`)；第二种，使用 jQuery 对象的一些方法。两种方式可以联合使用。

我们先将 1204.js 的内容修改为：

```
$(document).ready(function() {
    $("#orderedlist").addClass("red");
});
```

运行 http://127.0.0.1/1204.htm，相对于图 11-6 的变化部分截图如图 11-7 所示。

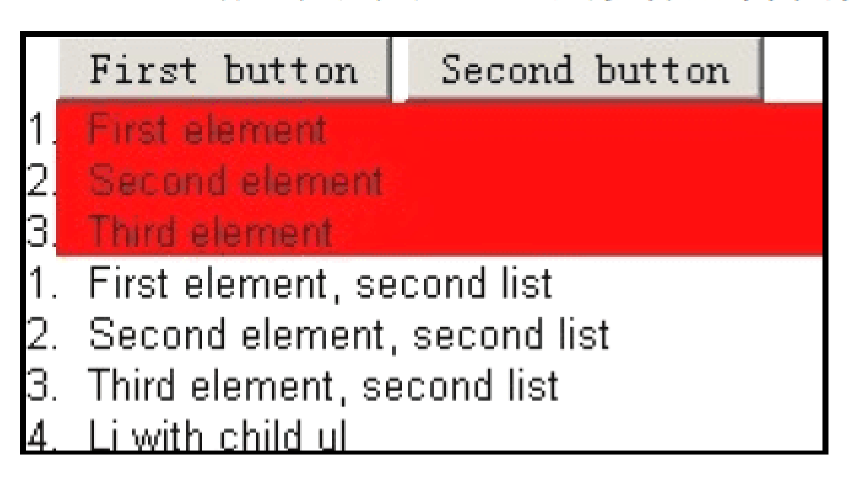


图 11-7　jQuery 效果 1-红色背景

图 11-7 与图 11-6 相比，部分内容(列表)的背景变成了红色，这是“`$("#orderedlist").addClass("red");`” 所起的作用。

`$("#orderedlist")`选择了 ID 为 orderedlist 的对象，在传统的 JavaScript 中，可以通过 `document.getElementById("orderedlist")`来选择。

`addClass("red")`表示 class 为 red，为对象添加一个红色背景的 stylesheet。

我们再将 1204.js 的内容修改为(修改 list 子元素)：

```
$(document).ready(function() {
    $("#orderedlist").addClass("red");
    $("#orderedlist > li").addClass("blue");
});
```

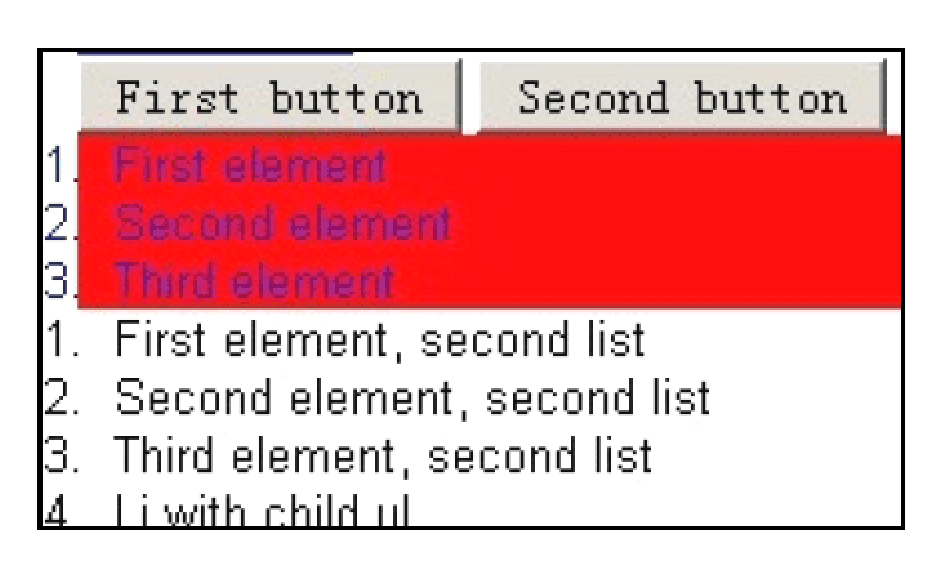


图 11-8　jQuery 效果 2-蓝色字体

如图 11-8 所示，已选择 id 为 orderedlist 的元素的所有子元素 li，并为其添加 class "blue"，即列表的三个子元素的字体变成了蓝色。

现在再看看一个更复杂的效果：当用户 hover(鼠标移到某处)到该 list 的最后一个 li 元素时添加改变颜色，否则不变。

我们可修改 1204.js 的内容为：

```
$(document).ready(function() {
    $("#orderedlist li:last").hover(function() {
        $(this).addClass("green");
    },function() {
        $(this).removeClass("green");
    });
});
```

运行效果：当鼠标在内容“Third element”上时，颜色变为绿色，否则还是黑色。如图 11-9 所示。

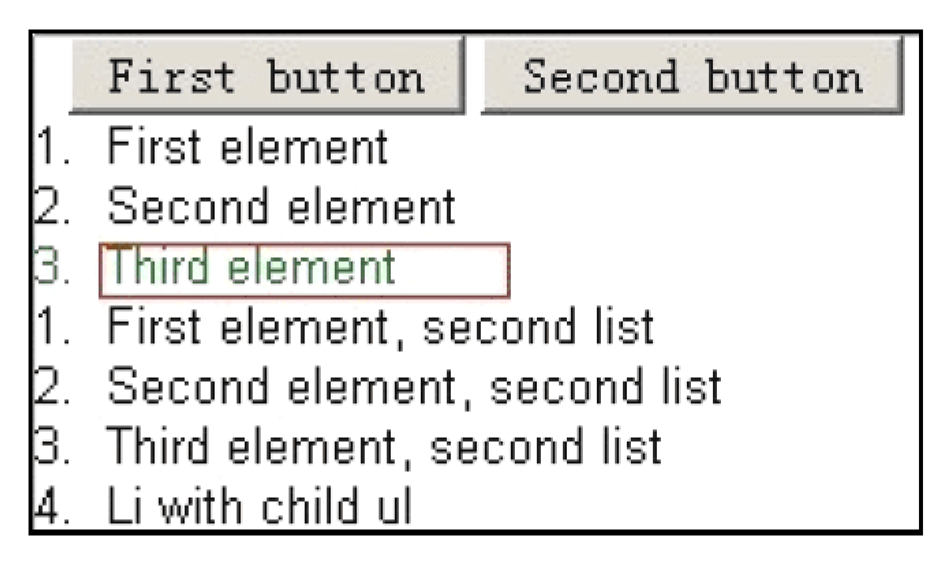


图 11-9 jQuery 效果 3-绿色字体

❖ **说明：**

1) 链接 http://api.jquery.com/category/selectors 拥有更多与 jQuery 选择器相关的知识及例子，请自行阅读。

2) 对每个可得到的 onxxx 事件，如 onclick、conchange、onsubmit，都有一个相应的 jQuery 等价物，详情请参考 http://api.jquery.com/category/events，这里有完整的事件列表，前面提到的 ready 和 hover，用来作为方法以方便某些功能的实现。

3) http://127.0.0.1/demo/jquery/selector.htm 示范了更多和 selector 相关的功能。

4) 对 jQuery 的学习也可参考 http://jquery14.com，号称 14 天就可学好 jQuery。但本书的观点还是“学以致用”，很少用到的知识点根本就不用浪费太多的精力去学习。

使用 selectors 和 events 已经可以做许多事情了，但是还有一个更有趣的用法。

我们将 1204.js 的内容修改为：

```
$(document).ready(function() {
    $("#orderedlist").find("li").each(function(i) {
        $(this).append(" BAM! " + i);
    });
});
```

运行效果如图 11-10 所示。

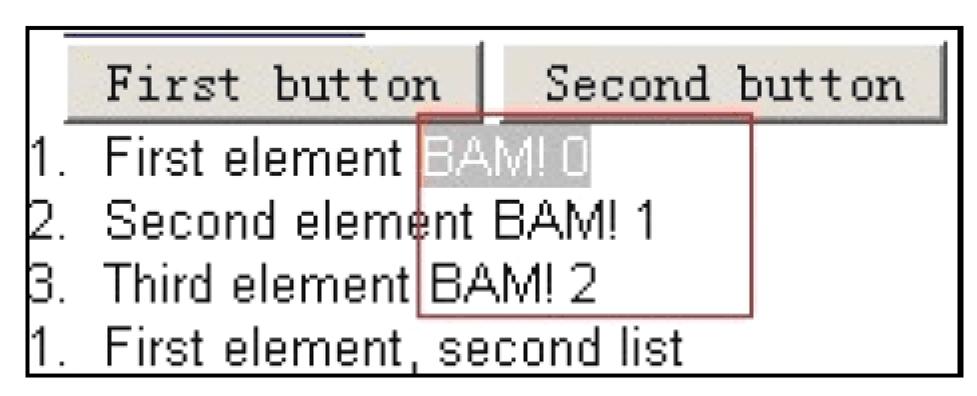


图 11-10 jQuery 效果 4-列表子元素添加内容

find()允许对已经选择的元素做更深入的搜索，$("#orderedlist").find("li")几乎和$("#orderedlist li")一样，但 each()对每个元素迭代并允许更多的处理，append()在每个元素的末尾添加一些文本。

另一个可能面临的问题是不选择某一个元素，jQuery 提供了 filter()和 not()。filter()通过满足该 filter 表达式来减少选择的元素；not()则相反，去掉所有满足该表达式的元素。

我们将 1204.js 的内容修改为：

```
$(document).ready(function() {
    $("li").not("[ul]").css("border", "1px solid black");
});
```

运行效果如图 11-11 所示。

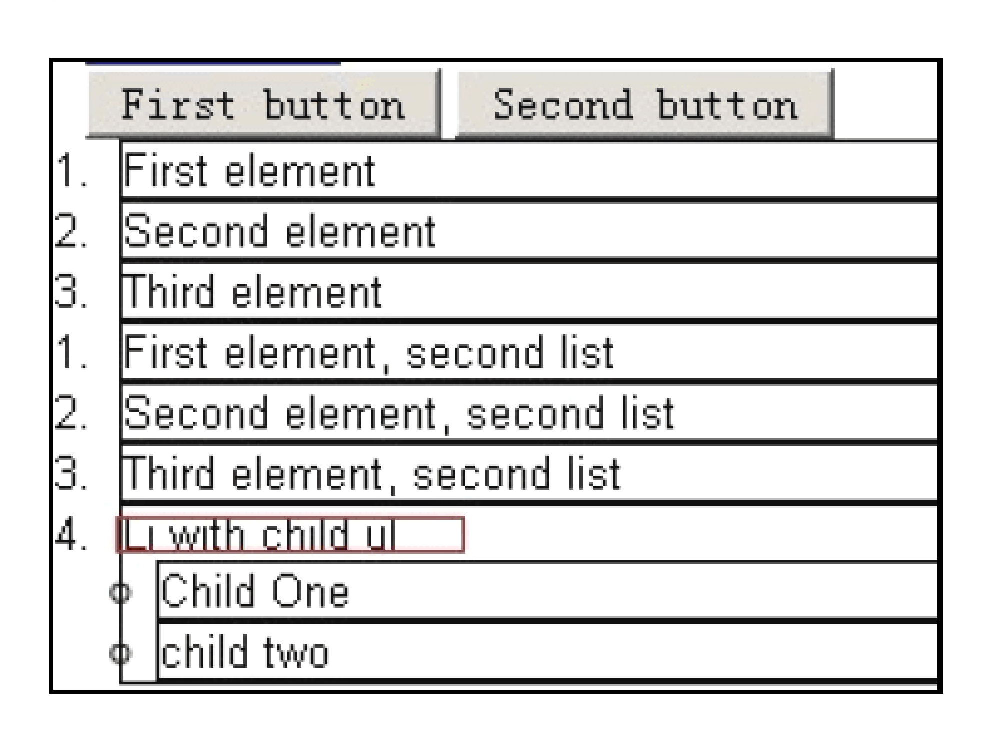


图 11-11 jQuery 效果 5-排除有 ul 的列表子元素

$("li").not("[ul]")选择所有的 li 元素，但有 ul 子元素的元素排除在外，故除了有 ul 子元素的其他 li 元素，都有一个 border。[expression]语法来自 XPath 并且可以用来被子元素和属性 filter。

2. 简单 Ajax 实现

下面通过一个小 Ajax 程序，允许用户参与评分，用服务器端程序 1205.php 来完成这个功能，该文件读 rating 参数并返回评分次数和平均评分。

我们将 1204.js 的内容修改为：

```
$(document).ready(function() {
    // generate markup
```

```
        var ratingMarkup = ["Please rate: "];
        for(var i=1; i <= 5; i++) {
            ratingMarkup[ratingMarkup.length] = "<a href='#'>" + i + "</a> ";
        }
        var container = $("#rating");
        // add markup to container
        container.html(ratingMarkup.join(''));

        // add click handlers
        container.find("a").click(function(e) {
            e.preventDefault();
            // send requests
            $.post("1205.php", {rating: $(this).html()}, function(xml) {
                // format result
                var result = [
                    "Thanks for rating, current average: ",
                    $("average", xml).text(),
                    ", number of votes: ",
                    $("count", xml).text()
                ];
                // output result
                $("#rating").html(result.join(''));
            });
        });
    });
```

说明：先用 jQuery 生成 5 个 anchor 元素，之后把它们添加到 id 为 rating 的容器元素内，并对容器里的每个 anchor 添加一个 click 处理器。当点击 anchor 时，一个以 anchor 的内容为参数的 post 请求发送到 1205.php。结果作为一个 XML 返回，并添加到容器中代替 anchors。

如果没有安装 PHP，也可看看 http://jquery.bassistance.de/example-rateme.html 提供的在线范例。

运行效果及点击链接后的效果分别如图 11-12 和图 11-13 所示。

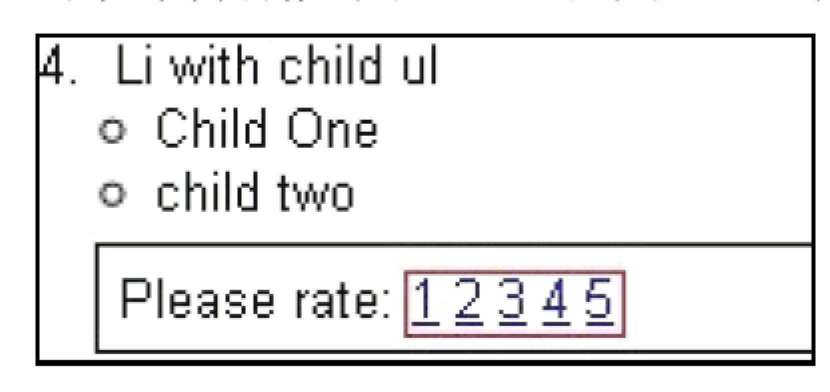


图 11-12　jQuery 与 Ajax

4. Li with child ul
- Child One
- child two

Thanks for rating, current average: 4.57, number of votes: 7

图 11-13　jQuery 与 Ajax 单击链接后的界面

1205.php 的代码如下：

```
<?php
define('STORE', '1205.dat');

function put_contents($file,$content) {
    $f = fopen($file,"w");
    fwrite($f,$content);
    fclose($f);
}

if(isset($_REQUEST["rating"])) {
    $rating = $_REQUEST["rating"];
    $storedRatings = unserialize(file_get_contents(STORE));
    $storedRatings[] = $rating;
    put_contents(STORE, serialize($storedRatings));
    $average=round(array_sum($storedRatings)/count($storedRatings),2);
    $count = count($storedRatings);
    $xml = "<ratings><average>$average</average>
            <count>$count</count></ratings>";
    header('Content-type: text/xml');
    echo $xml;
}
?>
```

代码说明：

1) 先定义文件名常量 STORE，再定义文件操作函数 put_contents。

2) 先检查参数 rating 是否已定义，若已定义，则读取文件内容并序列号，之后再写入文件。

3) 计算平均评分及评分次数，最后返回平均评分和评分次数的 xml 包。

3. jQuery 函数

jQuery 1.4 已经发布，相对于 1.3 版，性能又有很大提高，并显著降低了大部分热门 jQuery 方法的复杂度，使 jQuery 的使用更加简单。对于 jQuery 的使用，大多通过其提供的函数来实现。函数列表及详细信息可从 http://api.jquery.com/获得，在此不详细说明，仅

举“下拉菜单”来说明几个函数的用法。

先看图 11-14 所示的效果图：



图 11-14 jQuery 下拉菜单

请参考 D:\howwe\wwwroot\1206.htm，CSS 为 1206.css，请自己去阅读。下面仅保留 jQuery 代码部分，解释在相应的代码之后：

```
$(document).ready(function(){
  $("#nav-one li").hover(//id 为 nav-one 的子元素 li 触发事件
    function(){ $(this).fadeIn("fast"); }
    //将 ul 和悬停时的 li 变为对象，并启用 fadein 函数，将它显示出来
    ,function() { }
    //移出时无操作
  );
  if (document.all) {
    $("#nav-one li").hoverClass ("sfHover");
    //将 nav-one 的子元素 li 产生.hoverClass 方法，具体内容声明如下
  }
});

$.fn.hoverClass = function(c) {//声明了一个方法，这里的 c 代表传参
  return this.each(function(){//对象迭代
    $(this).hover( //对象悬停事件
      function() { $(this).addClass(c);  },//添加一个类
      function() { $(this).removeClass(c); }//移除一个类
    );
 });
};
```

当然也可以将`$("#nav-one li").hoverClass ("sfHover")`直接写成以下代码：

```
$("#nav-one li").hover(function(){$(this).addClass("sfHover"); }
,function(){ $(this).removeClass("sfHover");});
```

分开写是为了让代码更清晰，并了解如何在 jQuery 中自定义方法及调用，充分体会 jQuery 的强大！

4．编写 jQuery 插件

为 jQuery 写插件非常简单。若能遵循以下规则，其他人就能很容易地集成你提供的插件。

1) 插件命名

为你的插件选择一个名称，如 foobar，然后创建一个 jquery.[yourpluginname].js 文件，如 jquery.foobar.js。

2) 添加自定义方法

通过扩展 jQuery 对象来创建一个或多个插件方法，如：

```
jQuery.fn.foobar = function() {
    // do something
};
```

通过执行下面代码就可以使用上面的插件：

```
$(...).foobar();
```

3) 默认设置

创建可以被用户更改的默认设置，例如：

```
jQuery.fn.foobar = function(options) {
    var settings = jQuery.extend({
        value: 5, name: "pete", bar: 655
    }, options);
};
```

你可以直接调用该插件，即不带选项，这将启用默认配置。

不带选项调用：

```
$("...").foobar();
```

带选项调用：

```
$("...").foobar({ value: 123, bar: 9 });
```

4) 文档

如果要发布自己的插件，最好能提供一些例子和文档，以便读者了解及使用。让我们来写一个我们自己的插件。

有关插件的信息，可以参考现有的优秀插件。

范例：http://gmap.nurtext.de，gMap is a lightweight jQuery plugin that helps you embed Google Maps into your website。(gMap 是一个轻量级 jQuery 插件，可以帮助你将 Google 地图嵌入到你的网页中。)gMap 是“Google Maps Plugin for jQuery”的简称，1.1.0 版的只有

7KB。使用很简单，见范例 1207.htm，重要代码如下：

JS 引用及调用部分：

```
    <script type="text/javascript"
src="http://maps.google.com/maps?file=api&amp;v=2&amp;sensor=false&amp;key=
ABQIAAAA6cQIrMEc9zlaKBjWiPM5rxSjlBXfTSDcGsB79vzL90uiOHMpbBRa1FFoX2YfuQNFvFK
xQtpz0ZCeuw&amp;hl=de"></script>
    <script type="text/javascript" src="jquery.min.js"></script>
    <script type="text/javascript" src="jquery.gmap-1.1.0.js"></script>

    <script type="text/javascript">
    $(function()
    {
        $("#map").gMap();
    });
    </script>
```

HTML 部分：

```
    <div id="map" style="width: 547px; height: 320px; border: 1px solid #777;
overflow: hidden;"></div>
```

http://gmap.nurtext.de 的页面截图如图 11-15 所示。运行 http://127.0.0.1/1207.htm，截图类似于图 11-15。

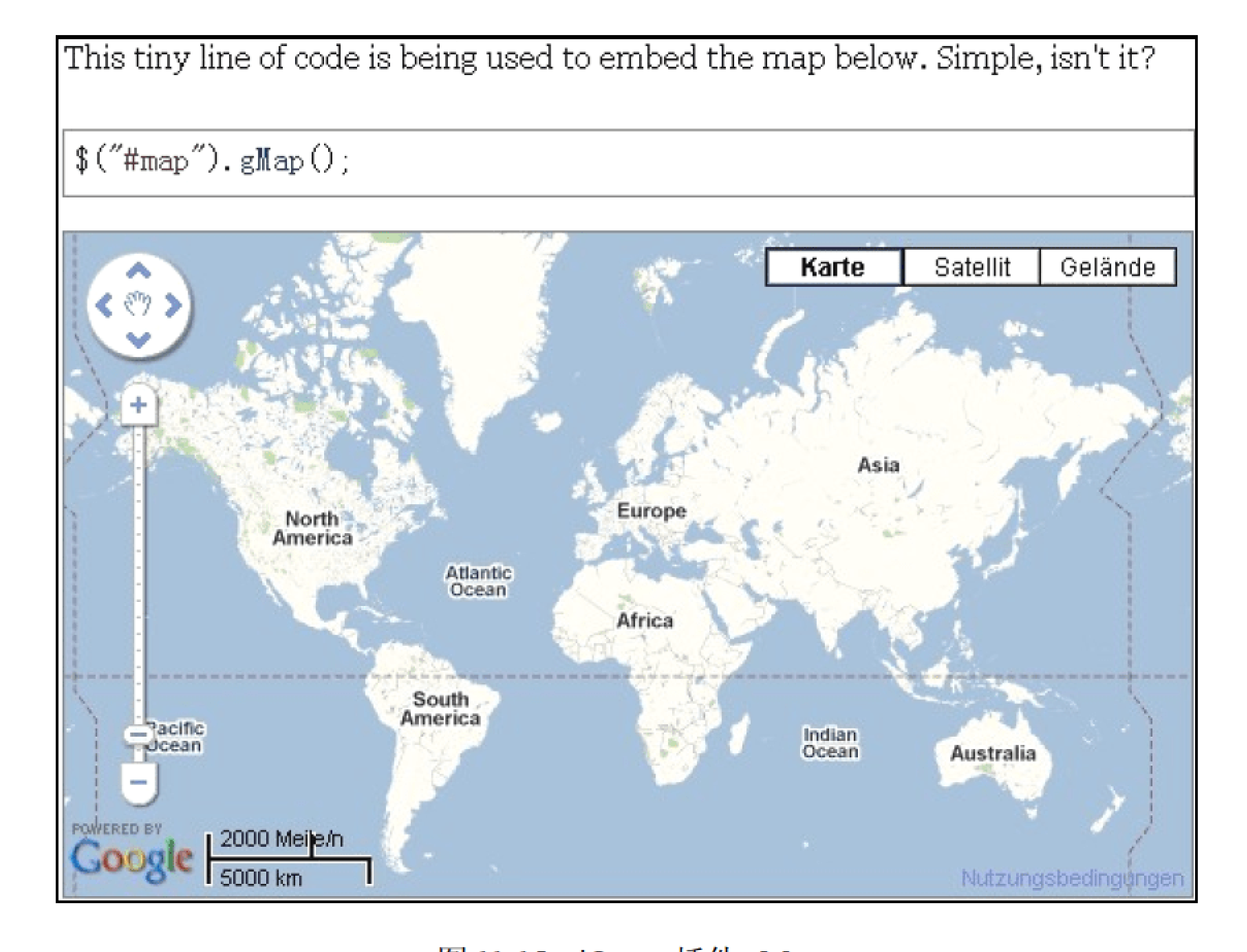


图 11-15　jQuery 插件 gMap

11.4　基于 jQuery 的门户框架——JPolite

JPolite 是一个基于 jQuery 开发的轻量级门户(Portal)框架，可用于创建个性化页面。JPolite 将内容、展示和事件相分离，可以很方便地将内容模块转换成基于标签(Tab)或折叠面板(Accordion)的方式进行展示。开发人员可以利用各种服务器端技术和框架，迅速创建门户网站。

源码下载地址为 http://code.google.com/p/jpolite2，提供基于 jQuery 1.32 和 1.40 的两个版本。http://code.google.com/p/jpolite 为 JPolite 前一版本，可对比效果，示范网址为 http://www.trilancer.com/jpolite2。

运行 http://127.0.0.1/jpolite，代码位置为 D:\howwe\wwwroot\jpolite，部分截图如图 11-16 所示。

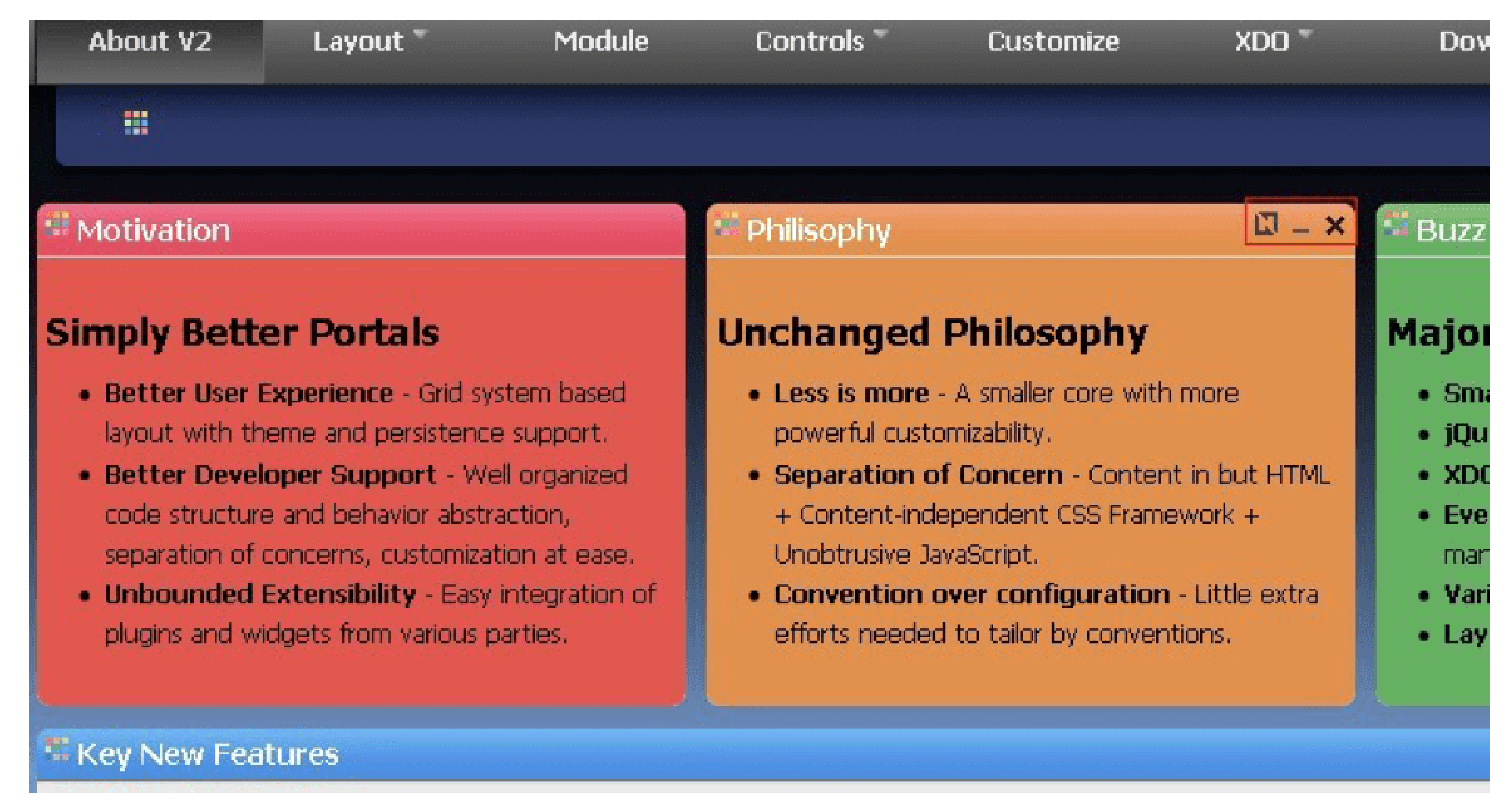


图 11-16　JPolite 运行截图

❖ 注意：

将鼠标移到名为“Philisophy”的窗口，窗口右上角将出现三个按钮；名为“Motivation”的窗口右上角无按钮。其实，该页面有 4 个窗口，分别是：Motivation、Philisophy、Buzz、Key New Features。但每个窗口都可以移动、刷新、缩小或关闭，单击深蓝色横条右上方的三角还可以将 4 个窗口都最小化，单击倒三角则恢复常态。更多操作可参照提示随意进行。

以下是 JPolite 原理的自述，选择菜单：**XDO -> Programming Model**，图 11-17 为其原理图。JPolite 根据 js\modules.js 对页面内容、布局等的描述，将相应页面加载到主页 index.html 上。详细原理可浏览 js 目录下的 modules.js、jpolite.core.js 和 jpolite.ext.js。

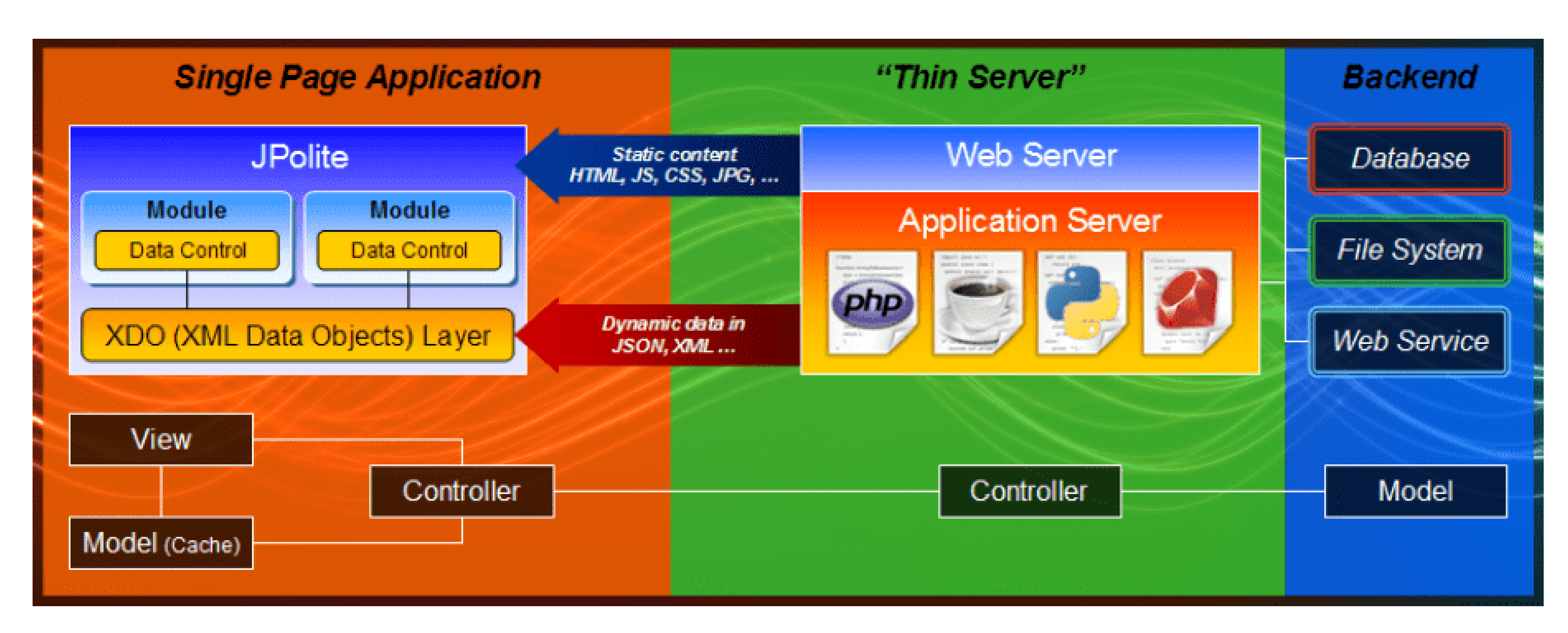


图 11-17 JPolite 原理图

再以首页为例，简单介绍 JPolite 的使用：

index.html 的内容按顺序分为：

- 对 CSS 的引入。
- id 为 header 的层所包含的菜单(id 为 main_nav)及信息栏(id 为 info_bar，深蓝色横条)。
- id 为 content 的层所包含的 4 个窗口，前 3 个窗口是加载的，文件分别为 m101.html、m102.html 和 m103.html。
- id 为 footer 的层所包含的页脚。
- class 为 module_template 的三个模板。
- 对 js 的引入。

为了能加载上述三个 HTML 文件，index.html 中有如下定义：

```
<div id="c1" class="cc"></div>
<div id="c2" class="cc"></div>
<div id="c3" class="cc"></div>
```

js\modules.js 中有如下定义：

```
var _modules={
m101:{url:"modules/m101.html",  t:"Motivation"},
m102:{url:"modules/m102.html",  t:"Philisophy"},
m103:{url:"modules/m103.html",  t:"Buzz"},

var _moduleLayout={
t1:["m101:c1:red", "m102:c2:yellow", "m103:c3:green"],

var _columnLayout = {
_default: { bg:'normal',
            c1:'span-8',
            c2:'span-8',
            c3:'span-8 last',
```

```
                c4:'span-24'
    },
```

index.html 中还有如下一个定义：

```
    <li id="t1"><b class="hover">About V2</b>About V2</li>
```

其实仔细研究_columnLayout 之后，很快就会发觉，_default 定义为 t1 更恰当。可见为了显示菜单“About V2”相关的页面，JPolite 先读取其 id：t1，再读取 modules.js 中 var _moduleLayout 和_columnLayout 与 t1 相关的内容，最后加载相应的页面(注：可加载的最大页面数为 4)。

其中：

1) `m101:c1:red` 的 m101 为页面参数，c1 为层名，red 为层(也可称为窗口)的背景。

2) `m101:{url:"modules/m101.html",t:"Motivation"}`的 url 针对具体的页面，t 为标题。

3) `c1:'span-8'`中的 span-8 为 CSS 参数。

❖ **注意：**

JPolite 的原理只点到为止，仅列出一条可供思考及深究的思路。

将_columnLayout 中的_default 修改如下：

```
     default: { bg:'normal',
                c1:'span-8',
                c2:'span-16 last',
                c3:'span-24 last',
                c4:'span-24'
    },
```

运行后部分内容的截图如图 11-18 所示。

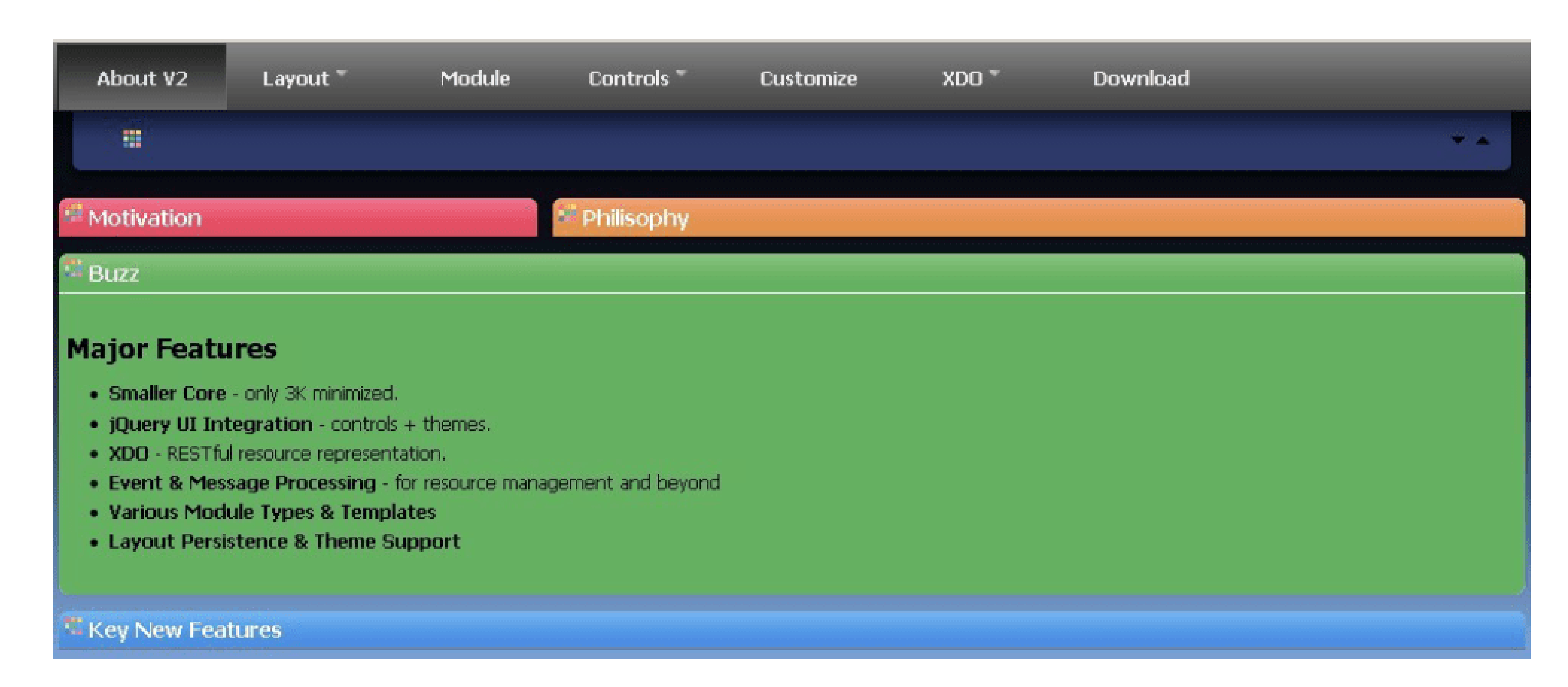


图 11-18　JPolite 修改布局效果图

11.5 jQuery 实战：HoCAS 表示层原理

有一天，突然看到 Google 的 Gears，发现 HoCAS 在表现层与 Gears 有相似之处。简单来说，HoCAS 表示层仅仅是在普通应用的基础上加了一个数据层(Data Layer)，如图 11-19 和图 11-20 所示。

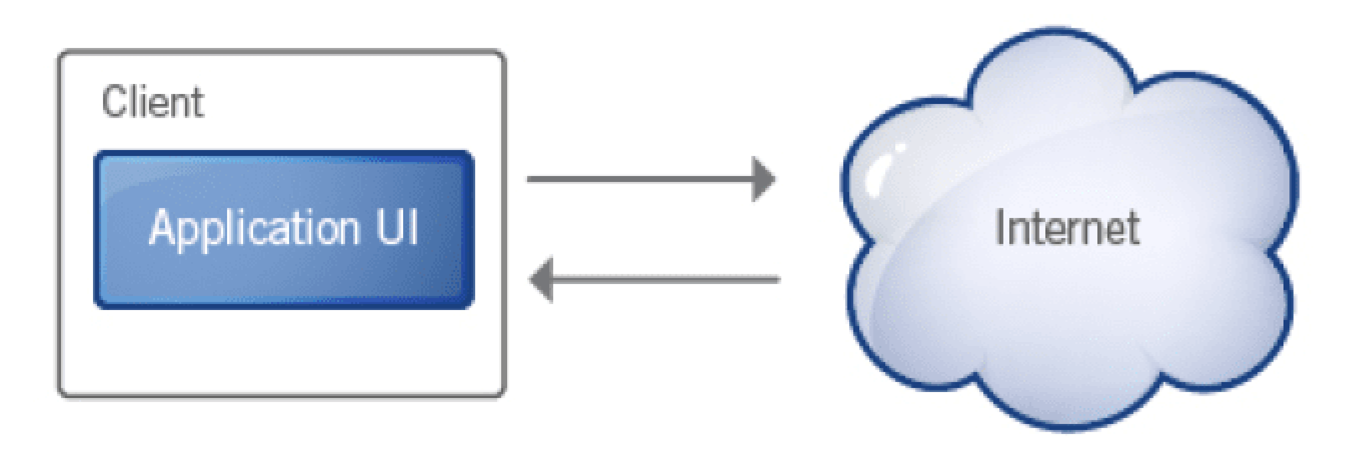


图 11-19 应用无数据层

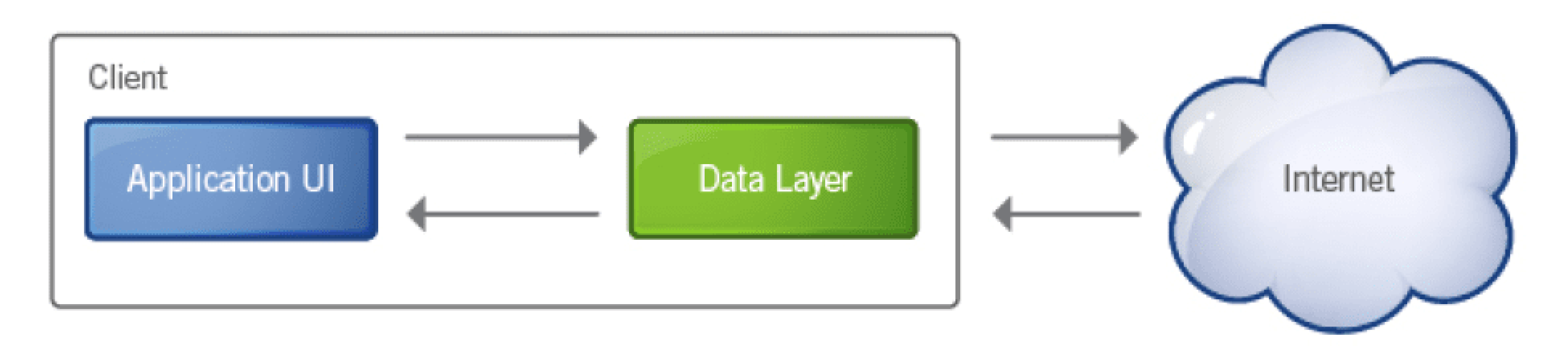


图 11-20 带数据层的应用

HoCAS 是 Howwe Configuration Application Server 的简称，中文名为浩为应用服务器。开发 Java 项目时，简单配置就能快速实现功能。即使是从不会写代码的人，简单培训，就能迅速配置出一个应用。

HoCAS 体现了软件即平台的思想。也就是说，软件不只是一个特定功能的定制产品，而且是一个能根据业务需求随时调整功能的应用平台。基于此平台，一个企业不仅能随时扩展自己的应用，而且还可以整合现有的应用系统。

简单一点说，HoCAS 表示层由数据模型和数据操作构成，界面由数据操作根据数据模型而生成。采用 jQuery 来生成 UI，JavaScript 定义数据模型。

Web 应用简单来说，其实就是对一堆数据进行处理，一般由多个页面构成，通过菜单来调用页面。而页面和数据相关联，数据则存放在表中。

HoCAS 先配置表存放数据、用页面和数据(表)关联，再用菜单关联页面，从而完成工作。这样，既可以根据业务的需要随时修改页面内容，而不用请人重新开发；又可以随时添加页面，甚至很容易就能把其他系统的数据整合到现有系统中。

一切就这么简单！

表、页面的配置数据还可以重复使用，以减少工作量。

HoCAS 既可用于需求分析时的快速建立模型，也可作为系统开发的基础。

HoCAS 拥有多种页面模板，对配置后生成的页面中的 js 做简单修改，就能完成新的

功能。图 11-21 是对 HoCAS 的简要说明。

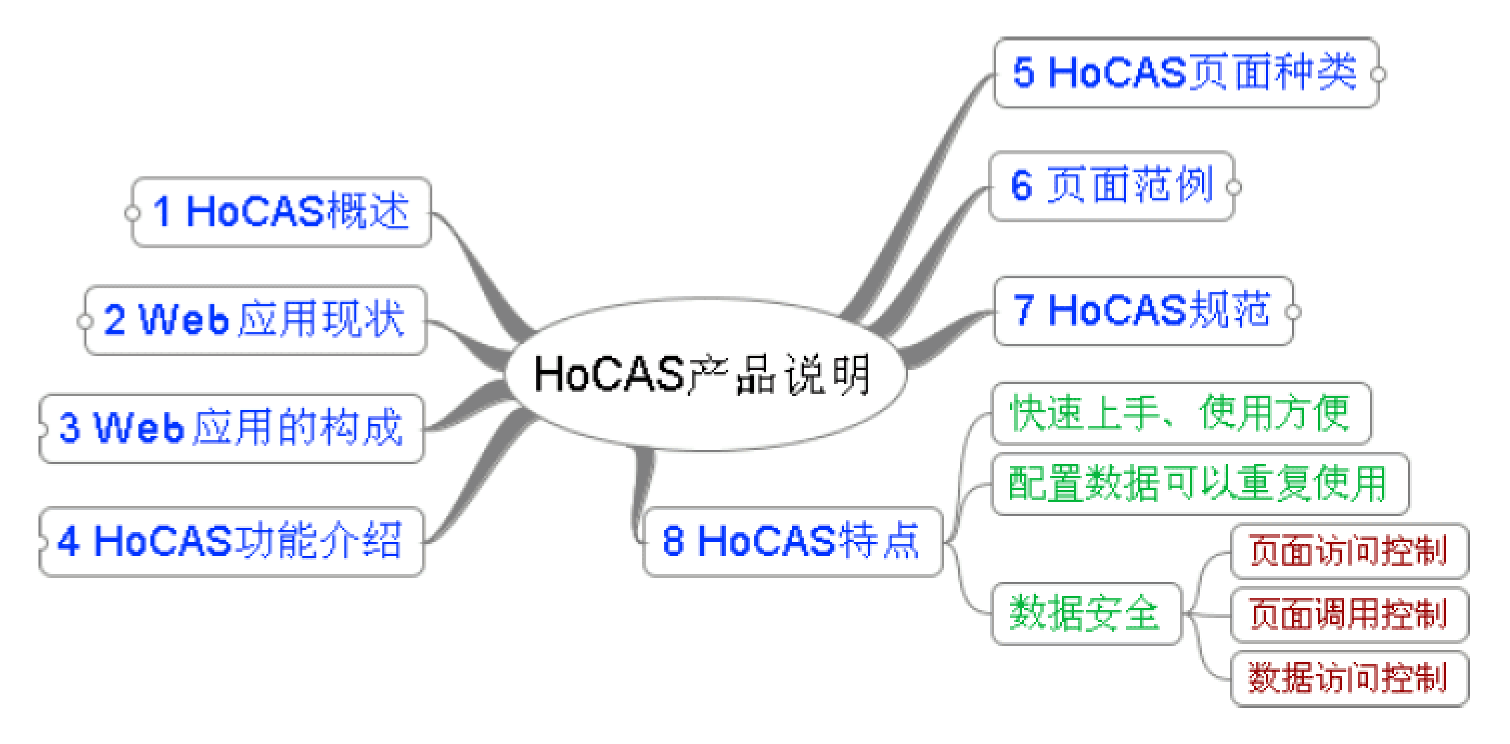


图 11-21　HoCAS 产品说明

详细的理论在此不想讲述，下面举例来简单说明一下。

图 11-22 为 HwPm 项目管理系统中**立项学校意见**的查询页面，功能是查询指定条件的项目。全部代码请浏览代码部分。

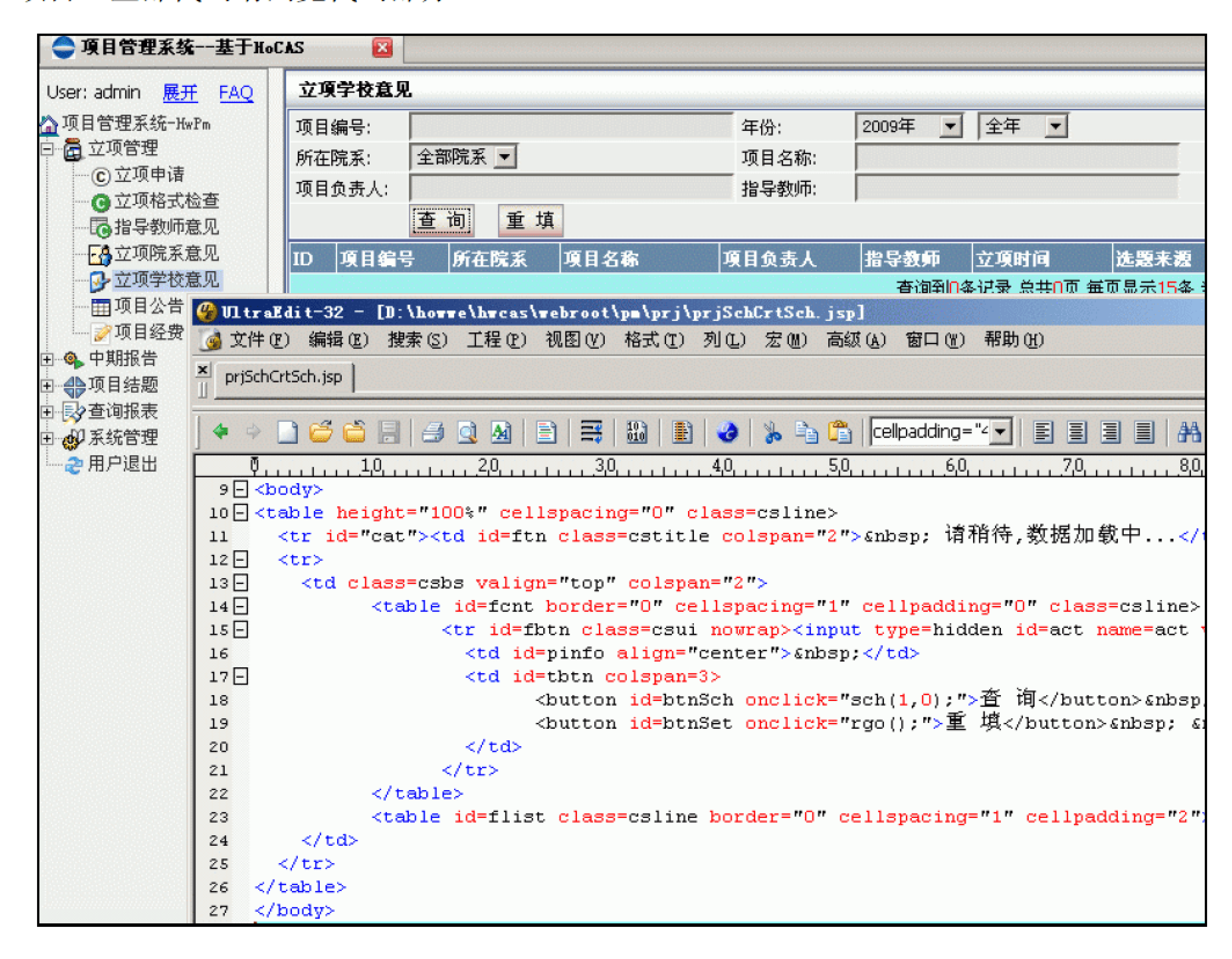


图 11-22　查询页面及 HTML 代码

图 11-22 所示网页的全部代码如下：

```
<%@include file="/img/common/pmheadsch.jsp"%>
```

```
<%@ page contentType='text/html; charset=GBK'%>
<html>
<head>
<title>.</title>
<meta http-equiv="content-type" content="text/html; charset=gbk">
</head>

<body>
<table height="100%" cellspacing="0" class=csline>
  <tr id="cat"><td id=ftn class=cstitle colspan="2">  请稍待,数据加载
      中...</td>
  </tr>
  <tr>
    <td class=csbs valign="top" colspan="2">
      <table id=fcnt border="0" cellspacing="1" cellpadding="0"
            class=csline>
        <tr id=fbtn class=csui nowrap><input type=hidden id=act name=act
            value=230><input type=hidden id=did name="did">
          <td id=pinfo align="center"> </td>
          <td id=tbtn colspan=3>
            <button id=btnSch onclick="sch(1,0);">查 询</button>   
            <button id=btnSet onclick="rgo();">重 填</button>     
          </td>
        </tr>
      </table>
      <table id=flist class=csline border="0" cellspacing="1"
             cellpadding="2"></table>
    </td>
  </tr>
</table>
</body>

<script LANGUAGE="JavaScript"><!--
function sch(page,rtl){
    var scond=urlsch(aui,"step=16")+"&flds="+recf(arh,2)+"&ord=did&fname=
        "+sload+"&sprjyear1="+$("#sprjyear1").val();
    schdb(page,rtl,scond);
}

$(document).ready(function(){
    aui.length=0;aui[aui.length]=['prjCrtSchChk','立项学校意见','1 查询',
               '0','3','geprj_info','prjCrtSch.jsp','',''];
    aui[aui.length]=['sprjno','项目编号','text','40','1','80','','',''];
```

```
    aui[aui.length]=['sprjyear','年份','select','40','2','40','','',''];
    aui[aui.length]=['smagdept','所在院系',
                        'select','40','1','40','','',''];
    aui[aui.length]=['lprjname','项目名称',
                        'text','40','2','80','','',''];
    aui[aui.length]=['sprjmag','项目负责人',
                        'text','40','1','80','','',''];
    aui[aui.length]=['lprjtch','指导教师',
                        'text','40','2','80','','',''];
    arh.length=0;arh[arh.length]=['idid','编号',
                        '80'];arh[arh.length]=['sprjno','项目编号','55'];
    arh[arh.length]=['smagdept','所在院系',
                        '55'];arh[arh.length]=['sprjname','项目名称','80'];
    arh[arh.length]=['sprjmag','项目负责人',
                        '70'];arh[arh.length]=['sprjtch','指导教师','55'];
    arh[arh.length]=['dcrttime','立项时间',
                        '70'];arh[arh.length]=['sprjfrom','选题来源','80'];
    arh[arh.length]=['sstep','步骤','60'];arh[arh.length]=['sflag','状态',
                        '30'];
    showui();
    $("#tdsprjyear").html(pmYearSct());
    gettbl();
});
function gettbl(){
    callpmui("&dact=getschdept&act=214&flds=name,name&cond=pid<2&ord=
                did&em=smagdept","script");
}
--></script>
</html>
```

❖ **提示：**

aui 与 auh 为数据模型，showui()用来显示页面，gettbl()用来获取相应数据。

代码说明：

1) 第 1 行包含头文件/img/common/pmheadsch.jsp，该文件定义了一些变量及功能函数(JS 函数，采用 jQuery 实现)。

2) 第 2 行指定页面编码为 GBK。

3) 第 3-27 行为页面的 HTML 代码。

4) 第 28-56 行为页面功能的实现，主要通过 JS 来加载数据。

整体思路：

在头文件中定义变量及功能函数，在第 37-46 行定义数据，通过“showui();”加载页

面内容，再用“gettbl();”加载数据并将数据在页面上显示出来。

如果要将上述页面改用 PHP 实现，只需将头文件 pmheadsch.jsp 改用 PHP 实现即可，基本不用对页面进行修改。而头文件其实就是功能模块，不管是用 Java 还是用 PHP 实现都不重要，重要的是功能模型如何定义。随着 HoCAS 的开源，将逐步提供 Java、PHP 等多种语言的实现，开源地址：http://code.google.com/p/howwe。

11.6 Gears 体系结构

Gears 的离线思想请参考图 11-23 所示的体系结构图：正常(网络连接正常)的 Web 应用在架构图中走的路线是沿着水平线横着与 Internet 通信。如果网络连接失败，用户依然能够访问，但 Data Switch 组件不再向 Internet 提交数据(即使提交也会失败)，而是向本地的客户端提交数据，将数据暂存在客户端本地的轻便型数据库中(Gears 使用 SQLite)，等到网络连接恢复时，再选择向 Internet 提交那些暂存的、待提交的数据。

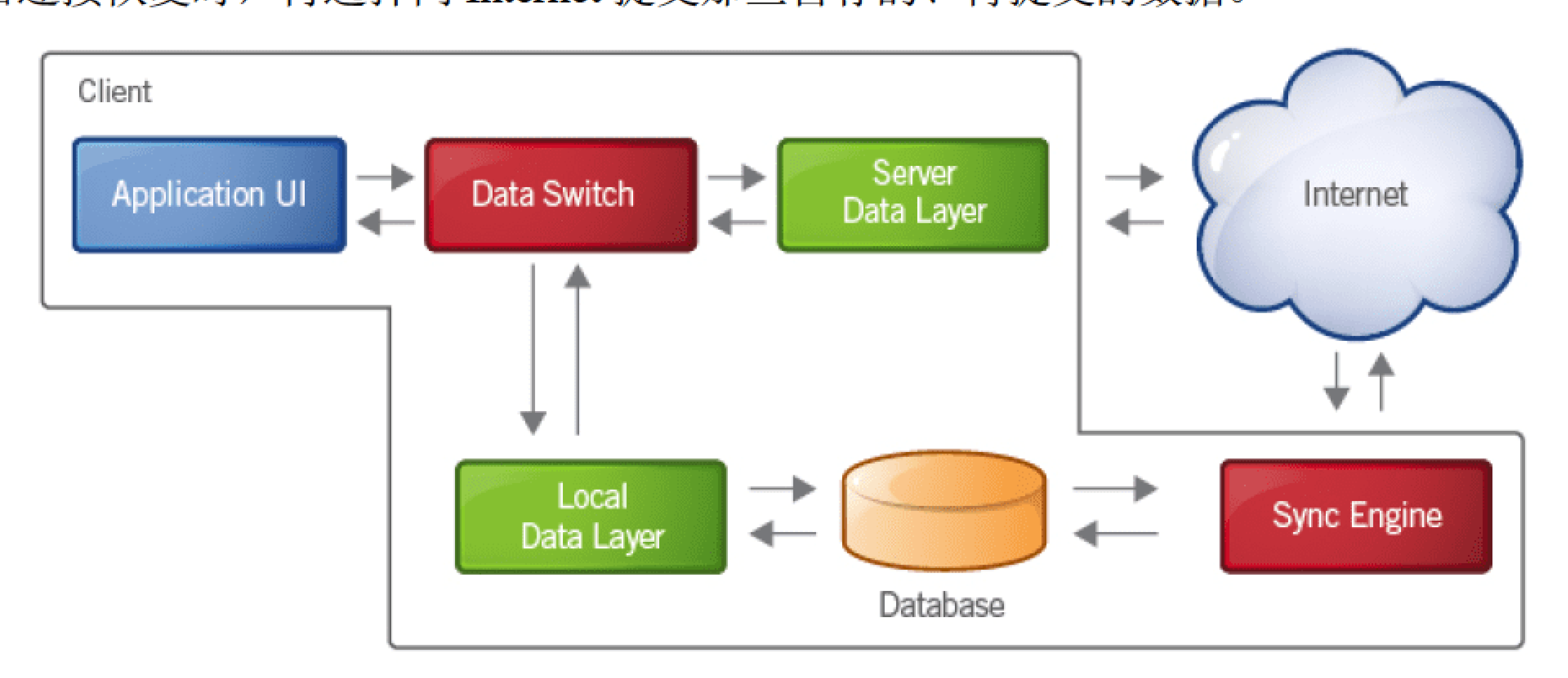


图 11-23 Gears 体系结构图

如果去思考，将产生一些疑问：

- Gears 用什么语言和 SQLite 交互？
- 网络恢复后再向服务器提交数据，意味着产生从 SQLite 中读取数据并删除等的 IO 操作，浏览器如何承受如此重的负担？
- 网络连接失败，真的依然能访问网页？

如果了解 Gears 的三大核心模块，就会明白 Gears 为啥能做到这些。

Google Gears 有几个主要的 API 组件：

- 一个本地服务器，用来存储和读取离线程序资源(包括 HTML、JavaScript、图像等)。
- 一个小型数据库(以 SQLite 构建)，用来存储本地数据。
- 一个工作池，用来让开发者将本地数据与服务器端数据同步。
- 一个桌面模型，可以使网络程序的操作贴近桌面程序。

- 一个地理定位模型，能够让网络程序侦测到目前用户的地理位置。

Gears 可以说是新形势下的老概念复活。这种模式跟 Ajax 一样，尽管形势明朗，但还有待时日的漫长转型期，将来定会拥有它的领地。

所谓新形势，是指 REST 架构的需求及 REST 可行性带来的机遇、SaaS 的需求、Ajax 培养起来的日益苛刻的用户体验需求和 Ajax 本身的局限性，加上 WPF 这些未来技术的压力，以及用户对它的迫切需求，势必寻找更现实的方案、更好的性能。

Gears 的提出，加上平台无关性、架构先进性和开放性，让广大开发者更有信心。

Gears 的提出，在 Web 开发上给架构带来很大冲击。如果简单地认为 Ajax 使传统 MVC 移了位置，而 Gears 则更需要去考虑 MVC 在什么地方、以什么粒度、何种方式重复，以及哪些数据需要离线存储、同步策略、冲突处理等，显然要求更敏捷、高效的工具，以及更精良的分析手段和架构。

就目前看 Gears 的定位与发展潜力，在 Ajax 与 RIA 的交接期间，纵横几年很正常；甚至凭着 Google 在互联网的成功及技术实力，Gears 迅速转变成 RIA 的老大，也都有可能。我们期待 Google 能创造一个奇迹……

Gears 是互联网向桌面融合的产物，而 RIA 是桌面向互联网融合的产物。详情请参考“11.8　Gears、AIR、WPF 的异同”。

11.7　Gears 离线应用原理

有关 Gears 信息的英文地址为 http://code.google.com/apis/gears/architecture.html，这里根据中文翻译只整理了部分内容。如果要看中文翻译，请自己去 Google，关键字：Gears、Architecture、翻译。

在 Gears 的发展过程中，Google 为了支持 Web 离线应用，研究过不同的结构，以下是各结构及其优劣的说明。

Web 应用要离线，必须注意以下问题：

- 分离的数据层。
- 连接策略，决定什么时候实现离线。
- 模态化，对应用程序的方式做出决定。
- 实现数据同步。

下面逐条介绍这几个问题。

1. 分离的数据层

1) 大多数 Web 应用没有真正的数据层(参见图 11-24)

Ajax 一般没有单独的通信层，通过代码直接调用。当只需要访问一个服务器的数据时，这种方式通常很不错。事实上，服务器提供给 Ajax 调用的 API 就是一个数据层。

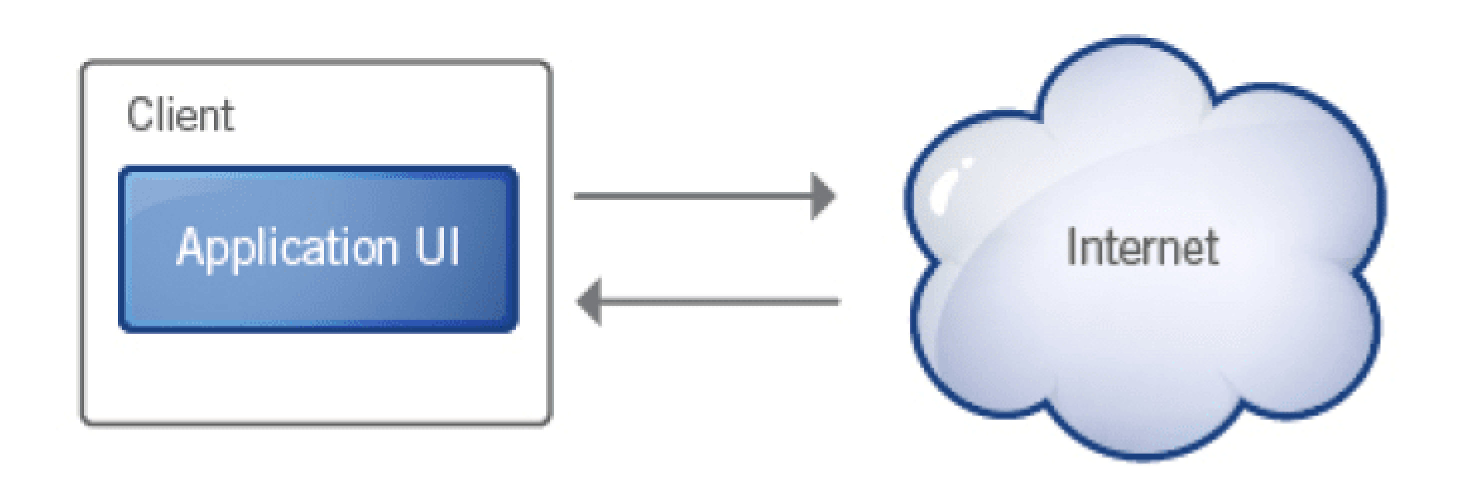


图 11-24　无数据层的普通应用

2) 数据层构架

通常，分离的数据层是实现优秀的第一步。

当为应用程序增加一个本地数据存储时(参见图 11-25)，就会有一个单独的位置，用来处理所有数据的存储及需要确认的请求。

例如：如果 Ajax 应用发起一个 JSON 请求到服务器，以获取一个用户的所有账户，则可以通过请求一个中间对象来获取用户的所有账户。这个对象可以决定是否从服务器、本地存储或两者的组合来获取数据。与之类似，当应用程序需要更新用户账户时，应用程序也只需要调用中间对象。中间对象可以决定是否将这些数据写入本地存储、是否发送数据到服务器，并且该对象能够定时同步。

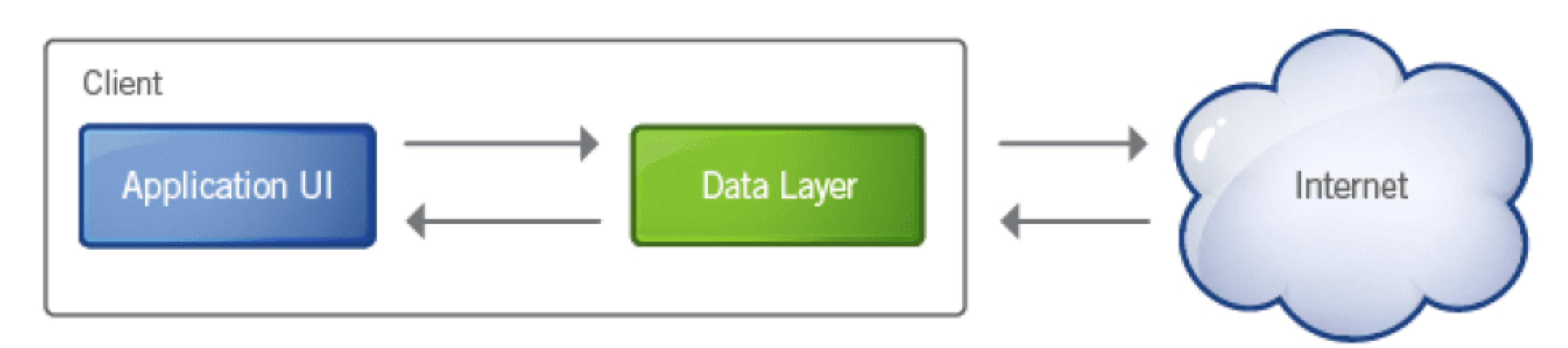


图 11-25　带数据层的应用

可以认为这个中间对象是一个数据交换层(参见图 11-26)，该数据交换层实现了与数据层一样的接口。第一步，可以使这个数据交换层将所有调用传递给数据层，该数据层直接和服务器交互。既然数据交换层的编码方式被遵循，当 Gears 没有安装或者用户不想让应用程序离线处理时，这一步将很有用。

❖ 注意：

数据交换层并非严格需要(例如 GearPad 就没有这样的数据交换层)。

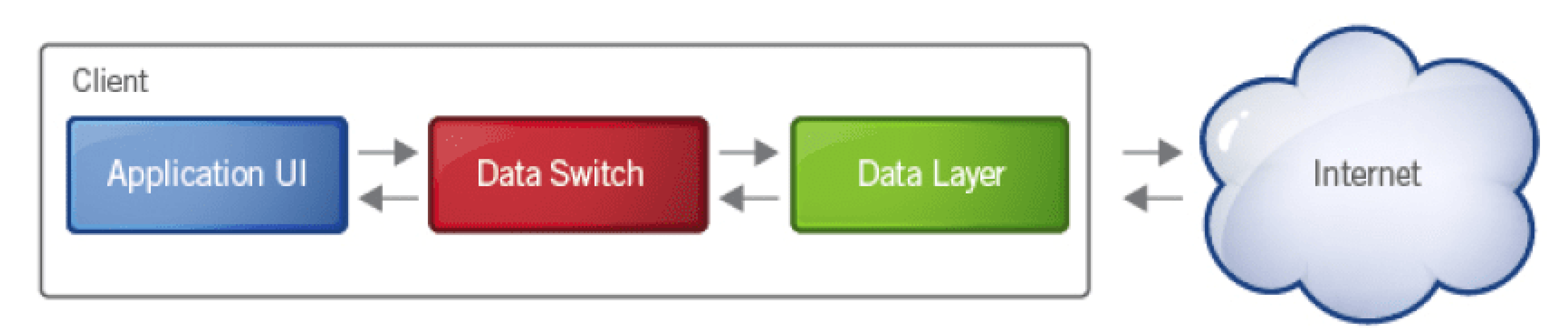


图 11-26　带数据交换层的应用

如图 11-27 所示，下一步将创建一个新的、使用 Gears 数据库的本地数据层，以代替从 Web 服务器获取数据。如果该数据层拥有这样的接口，那么这将变得非常简单，该接口和数据层原有用于与服务器交互的接口一样。如果接口有些不同，比如一些必须进行的转换，就在原有数据层处理。

为了测试这一步，可以设置数据交换层与本地数据层通信。不过，之前最好建立一个数据库，以便使测试更加容易。

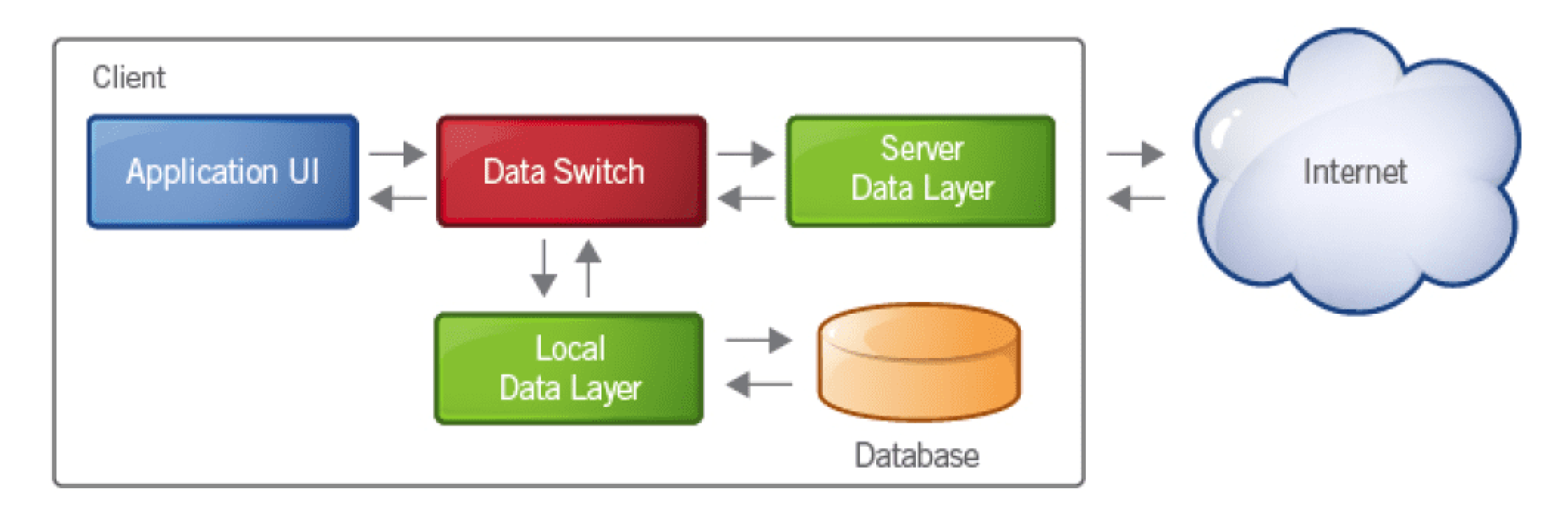


图 11-27　带本地数据层的应用

3) 无数据层架构

即便应用没有设计数据层并且没有必要添加一个数据层，也仍然可以通过在发起调用前拦截对 Web 服务器的调用而构建一个数据层。例如，可以监听 Submit 事件，在发起一个表单提交前拦截，并决定应用程序是使用本地存储的数据还是使用服务器上的数据。

实现这种方式需要找到所有发送请求到服务器的函数和方法，然后重新路由它们。面临的挑战是，这种方式需要很多额外工作，例如解析 URL 地址、像服务器返回结果那样重新生成表单。实际上，需要在客户端重复实现在服务器端所做的大部分工作。不管怎样，对于已经存在且不能重新设计架构的 Ajax 应用来说，这种方式是可取的。

2. 有效的脱机功能

出于实用的原因，应用程序的每一个功能不可能都成为可脱机使用的功能。所以，需要选择对哪些功能进行本地支持，并决定何时使用本地存储和何时连接服务器。我们将此称为“连接策略”。

你可能会认为总是会使用本地存储，因为它比较快。但是，也有很多实际的原因，可能反而想要或者需要访问服务器上的数据。例如：

- 数据太短暂，它在本质上没有必要去缓存。例如，可提供实时股票行情的应用，不会受惠于旧股票的报价服务。
- 有些应用的数据只有联网才有意义。一个极端的例子是，即时消息应用程序只有连接才有意义。

- 应用程序可以选择只存储访问最频繁的数据。例如，如果用户可以设置一个喜好，以改变应用程序的语言，则这种喜好的改变可能不支持脱机功能，因为与不改变喜好所带来的收益相比，执行这项脱机功能的成本较高，不值得这么做。
- 与计算和(或)磁盘空间相关的要求使创建脱机功能难以实现。例如，该功能需要大量的有用数据，超出了 PC 的合理数量。

典型情况下，最佳的解决方案是尽可能地使用本地存储，因为它通常快于远程连接。然而，应用程序在本地做的工作越多，就需要写越多的代码在本地执行该功能并做相应数据的同步。这时，就需要对成本和收益进行权衡，有些功能可能不值得本地支持。

3. 模态化

所有离线功能的应用程序都必须尽早回答的一个根本问题是“模态化”。

模态化应用程序有明显的联机与脱机模式，通常通过用户界面的一些变化来表示。用户使状态很明了并以一些方式来参与状态的转换。

无模态的应用程序在尝试无缝转换联机和脱机状态时，用户界面通常没有明显变化。用户不需要参加开关状态，应用程序会自动完成。

1) 模态应用

在一个模态应用中，应用程序联网时，它与服务器通信。当它脱机时，它使用本地存储。应用模态切换时数据必须同步。

实现模态化的应用程序的优点是，它的实现相对容易，并有一个合理的方式引导应用程序进入离线功能。

缺点是：

- 用户必须记得切换模态。如果忘记了，它们要么在脱机时没有所需数据，要么在无意断网时还联机工作。
- 如果网络连接时断时续，用户要么需要选择一个状态，要么在连上或断开时反复在两种模式之间进行切换。
- 由于本地存储并不总是最新的，不能用它来改善应用程序连接服务器时的响应能力。

2) 无模态应用

在无模态的应用程序中，应用程序运行于处在脱机状态的假设中，或者它可以在任何时间失去网络连接。应用程序尽可能地使用本地存储，并且当服务器可用时，持续在后台进行小的数据同步。当恢复联机时，数据同步也将进行。

无模态应用程序的优点是：

- 更好的用户体验、用户无需明白网络连接状况或转换状态。
- 即使网络连接时断时续，应用程序也将平稳顺利地运行。
- 由于本地存储动态更新，因而可用于优化服务器连接。

无模态应用程序的缺点是：

- 更加难以实现。
- 必须注意避免同步后台进程时占用太多资源，而使整个应用程序反应迟缓。
- 测试应用程序更有挑战性，因为同步逻辑发生在后台，而不是针对具体用户操作。

Gearpad 是一个无模态的脱机应用程序，它总写入本地数据库，然后独立地与服务器同步变化。

目前，Google Reader 是一个模态应用程序，它有一个独特的、用户必须显式启用的脱机模式。在“读者”状况下，进行状态的切换是一个务实的选择，因为它可以更快地实现，并借助 Gears 使 Google Reader 早期版本的释放成为可能。

4. 数据同步

无论使用的是哪种连接和模态策略，本地数据库中的数据都将与服务器中的数据完成同步。例如，本地数据和服务器数据完成同步时：

- 用户脱机时可更改数据。
- 数据被共享，并且可被外部各方修改。
- 数据来自外部源，如 feed。

解决这些差异以使这两种存储一样，称为“同步”。有许多方法可实现同步，但没有一个能完美地针对所有解决方案。最终选择的解决方案，将可能是为特定应用程序高度定制的。

下面是一些通用的同步策略。

1) 手动同步

最简单的同步解决办法就是“手动同步”，之所以称为手动的，是因为由用户决定何时同步。它可以简单地通过将所有旧的本地数据上传到服务器来实现，并在脱机前从服务器下载一个最新的副本。

手动同步的要求：

- 数据量足够小，以保证能在一个合理的时间内下载。
- 用户必须明确指出何时脱机，这一般通过用户界面的按钮来实现。

使用这种方法及脱机模式产生的问题是：

- 用户并不总是知道他们网络的连接状态。Internet 连接可能意外中断或时断时续，例如在公共汽车上。
- 用户可能脱机前忘记同步。
- 手动同步是很好的开始，因为它相对容易实现。但是，它需要用户在同步过程中有意识地参与。

2) 后台同步

在后台同步中，应用程序持续不断地在本地存储和服务器之间同步数据，参见图

11-28。可以通过下面方式实现，每隔一小段时间 ping 服务器，让服务器发送流数据到客户端(这在 Ajax 中被称为 Comet)。

这种同步方式的优点是：

- 所有时间数据都是准备好可以使用的，无论是用户选择离线还是突然断开连接。
- 当网速很慢的时候，可以最大程度地提高程序的性能。
- 下载的一方，当使用后台进程时(如果没有使用 WorkerPool 的话)，同步引擎可能会消耗资源或者降低在线体验度。使用 WorkerPool 的话，同步的开销将最小化，并且不会影响用户的体验。

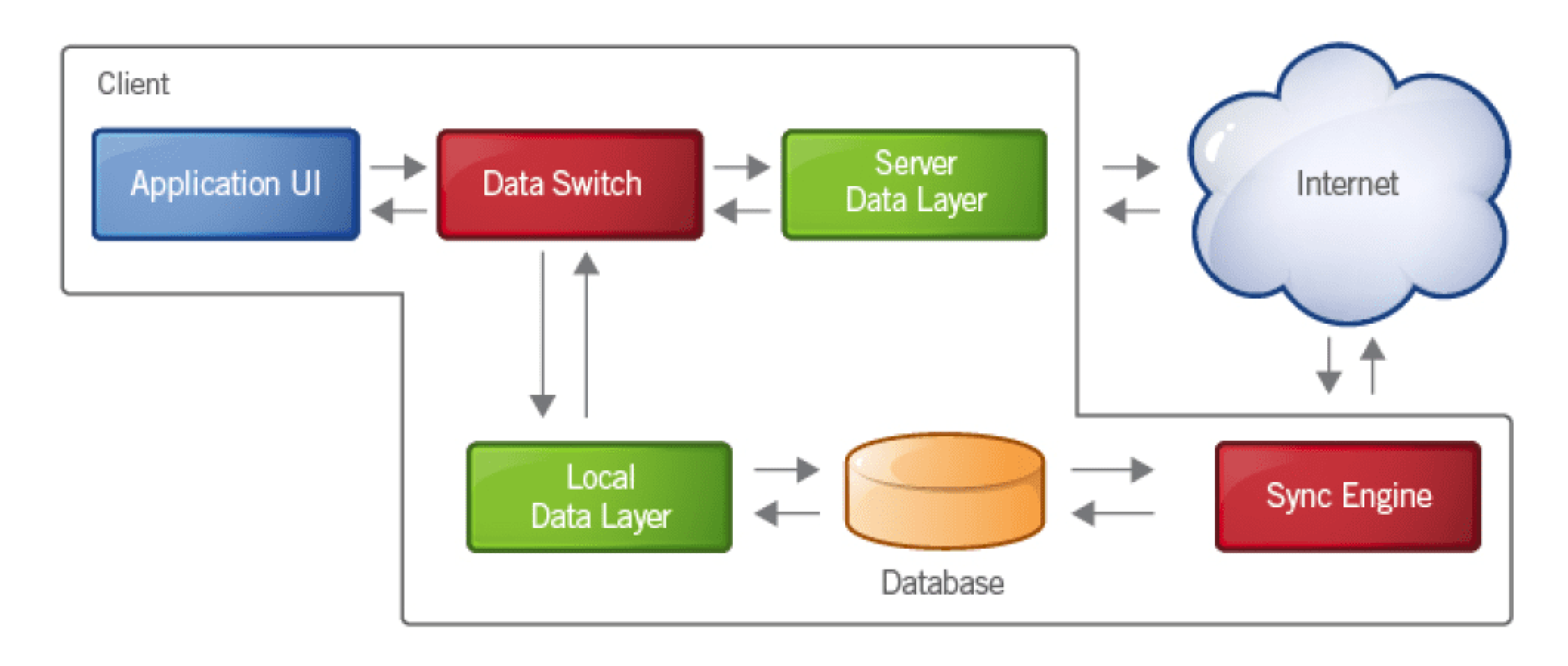


图 11-28 带后台同步构架的应用

总结

有许多种设计方案可实现应用程序的离线功能，这里回顾了少部分可选的、针对现状的通用设计方案。应用程序的选取，取决于想达到的用户体验以及应用的局限性和所要达到的目标。

11.8 Gears、AIR、WPF 的异同

Gears 是互联网向桌面融合的产物，AIR 与 WPF 等 RIA 产品是桌面向互联网融合的产物。

虽然 Google Gears、Adobe AIR/Flex、微软 WPF/Silverlight 等都是 RIA 解决方案，但却都是从各自优势出发而发展起来的技术方案。

1. 认清 RIA 及 Adobe 的 AIR

RIA 并非技术上的革命，只是商业上的革命。RIA 的技术封装让现有技术得到了极大的保留，设计师和开发工程师们可以方便地将自己的代码平移到新的平台上，并且发布、

部署的方式比之前的任何一款开发工具都方便得多。

其中 Adobe AIR 带来的革命，号称“Desktop 2.0”。其内容从 Flash、HTML/CSS/JS 到 PDF，几乎涵盖了时下最“流行”的 Web 内容载体。此外，“可离线”应用模式能让用户更加安全、舒适地进行工作和娱乐。用户不必再抱怨因网络故障而造成信息丢失，还可借助本地资源更好地节省带宽和其他“紧张而充满麻烦”的网络资源。

从商业角度来讲，AIR 创造的价值远胜于给用户带来的体验上的提升，对于终端用户来说，他们想要获得一份 Web-Desktop 应用，都要经过下载、安装、建立连接的过程，他们不会关心你的程序由什么开发而来。

从产品的角度讲，只要能进入桌面，有文件系统的支持以及本地各种服务、应用的支持，就不成问题。更多成熟的框架和接口，能让 UI 层直接调用 GPU 指令来构建绚丽的应用。

AIR 却让以往只有.NET/Java/C++程序员和企业才能达到的目标，对于一名普通的 Web Developer 或小型网站来说也“易如反掌”！多年以来，人们对于 Web 开发人员比.NET /Java/C++程序员差的观点势必减弱。抛开一些“浮躁”、“效率不济”等 Web 程序员和 Web 产品的“通病”不谈，Web 应用的优势是：其产品往往是各类软件产品中，面对用户群最多、最广、变动频率最高的产品。在优秀的 Web 产品中，用户体验已久经考验，尤其是此类产品往往还拥有较高的“用户体验开发效率”，即单位时间内用户的开发效率得以大大提升。

2. 什么是 AIR

AIR，Adobe Integrated Runtime 的简称，是一个跨操作系统的运行库。AIR 充分利用现有 Web 开发技术(Flash、Flex、HTML、JavaScript、PDF、Ajax)来构建富 Internet 应用程序(RIA)，并部署为桌面应用程序。

从本质上讲，AIR 提供了一个桌面与浏览器结合的平台，提供网络结合桌面应用的功能和丰富的表现方式。这对网络开发人员和桌面应用开发人员都极具诱惑力。

Adobe AIR 到底是个什么东西呢？先听听 Adobe 副总裁托马斯·黑尔(Thomas Hale)的解释：“假如你在使用谷歌地图的时候‘告诉’Web 应用程序，让它在你的房屋遭受飓风袭击之前通知你。如果你关闭了浏览器，谷歌地图就无法再与你‘交谈’了。而‘Adobe AIR’可以成为一个连接互联网与桌面应用程序的中间件。即便你关闭了浏览器，它还存在于计算机内存中，当 Web 应用通知该客户端你的房屋将要被飓风袭击时，它就会跳出来提示你。”

总之，Adobe AIR 让用户在自己所熟悉的环境下，利用最得心应手的工具，比如 Flash、Flex、HTML、JavaScript 和 Ajax，来建立和配置跨操作系统的桌面富互联网应用(RIA)。用户使用 AIR 应用程序的方式和传统桌面程序一样，当运行时环境安装好以后，AIR 程序就可以像其他桌面程序一样运行。

因为 AIR 是应用程序的运行环境，因此它很小并且对用户来说是不可见的。该运行环境提供了一套完全一致的、跨操作系统的平台和框架，用来开发和部署应用程序，因此你

的程序不必在每种操作系统上都进行测试，只要在一个系统上开发好了，就可以在其他操作系统上运行。这样有如下好处：

- 开发 AIR 应用程序不必做额外的跨平台工作，节省了时间，因为跨平台的工作 AIR 都帮我们做好了(只要其他平台能支持 AIR 即可)。
- 比起 Web 技术及其设计模式，AIR 应用程序开发迅速，它允许将 Web 开发技术搬到桌面上来，而不用另学桌面程序开发技术或复杂的底层代码，这比起低级语言(如 C 和 C++)更容易学习，并且不用去处理每个操作系统复杂的底层 API。

3. Adobe 情况简介

Adobe 利用 Macromedia 当年 FlashPlayer 积累的绝对优势来向外“辐射”AIR 应用：一方面，RIA 产品能做到无缝安装，体验流畅；另一方面，开发者资源是 Adobe 最大的优势 ——Web Developer(HTML、CSS、Ajax 等)和 ActionScript Developer 的结合是一股强大的势力，不容忽视。当年 Macromedia 被收购时，Adobe 的一位高管表示购买对象就是 Flash，其实 Adobe 购买的是 FlashPlayer 在大众计算机上极强的渗透力和 Macromedia 对开发者、设计师极强的社区黏度。

Adobe 的优势：

- 极大的 FlashPlayer 覆盖率，即便是初次安装 Adobe AIR，也能感觉到“无缝”：用户不会被强硬的“下载”过程所干扰。
- 绝大多数设计师都是 Flash、Photoshop 的忠实用户，让设计师改变使用习惯比让程序员改变使用习惯要困难得多，这恐怕是微软最头痛的地方。如果要招一个不用 Adobe Flash 做动画、不用 Adobe Photoshop 做设计的设计师，会非常困难。

4. 微软情况简介

微软是一部巨大的商业机器，似乎永远都不缺钱，也不缺赚钱的方法。这部赚钱机器投入到 RIA 应用，利用本身的平台优势进行“辐射”是非常迅猛的。在微软与一些互联网/传媒巨头(NBC、AOL、Nokia、Tencent 等)进行合作的大背景下，每天的装机量就远远超过 100 万次！

微软的优势：强大的商业机器、强大的资金支持以及得天独厚的平台支撑。毕竟现在 Windows 操作系统还是主流中的主流。你可以主观地讨厌它，但永远不能忽略它。

5. Adobe 与微软的策略

简单一点说，AIR 只是一个和 Ajax 组件包一个层次的解决方案，主要用于客户端表现。许多人没有明白这一点，却拿 AIR 和 Delphi、Java、.NET 等语言去比。其实 AIR 只专注客户端表现，你的业务层完全可以用 PHP、Java、C#等语言来写。

再说，AIR 技术的本质就是 HTML + JavaScript，只不过有些改进，换了个名字，叫 MXML 或 ActionScript。而微软的 WPF 技术，也只是这两个技术的组合，大家的原理都

一样。

Adobe 很聪明，知道微软这个长年想扩展到互联网却屡屡受挫的困兽，必然拼了老命也要推自己的 WPF 和 WCF 技术，因为这是微软未来的命。如果采用和微软一样的策略，想在这个霸主的肩膀上吃肉，比登天还难。所以 Adobe 祭出微软最害怕的一招：开源、免费、基于开放技术标准、跨平台。

Adobe 一直擅长客户端表现，但不擅长开发工具的开发，所以 Adobe 只提供了 AIR 这样的解决方案，而没有提供从服务器到客户端的一整套技术方案。

Windows Vista 本是微软最寄予厚望的操作系统，原生的.NET Framework 3.0，而且全部用.NET 重写，一流的面向服务的操作系统，圆了当年微软 COM 之梦。而且.NET Framework 3.0 最主要的技术——WPF、WCF，还有一个腹死胎中的 WF，都是未来的前瞻技术，由于技术难度太大，因为太多的人看透了微软在互联网战略的陌生和迟疑，以及微软固有的 EXE 技术思想，以至大都跑到纯正的互联网公司(如 Google)去了，使 Vista 难产。

微软本希望能无缝且毫无体验差异地整合互联网和本地桌面应用，但这个过程太漫长了。眼看着 Adobe 利用 Flash 做跳板，占据了自己未来想占领的位置，于是微软急匆匆推出了 Silverlight。本来微软就没有 Silverlight 的研究计划，希望借助 Vista 一举掀开未来大幕，但 Vista 不争气，只得用 Silverlight 应急。

Adobe 的 PDF 抓住了 PC 时代，Flash 抓住了浏览器时代，AIR 想去抓住浏览器和 PC 融合的时代。故 AIR 也是 Adobe 的命根，所以两家公司都拼了命。微软有其强大的现有客户和研究开发中心及营销手段，Adobe 则采用了最开放的联盟来对抗，让微软以一家公司之力来对抗全球开放的程序员。Adobe 的 AIR 还没有练到家，而 Flash 是 Adobe 的现有优势，要想达到预期目的，Flash 必成为 Adobe 的跳板，这也是 Adobe 用 Flex 作为过渡的原因。

在开发工具方面，微软有自己的 Visual Studio。但微软的对立阵营就有最开放的 Eclipse 及 NetBeans。Eclipse 不仅仅是 Java 的开发 IDE，Ruby、PHP、C++、Python 等都能在它上面运行。故 Adobe 也把自己插在 Eclipse 上面，跟着程序员大潮走，让程序员使用自己最熟悉的语言，通过 IDE 开发 AIR。

6. Google Gears 的优势

无论是 Adobe 还是微软，在战略定位方面，都远远落后于 Google。Google 目前是互联网老大，同时也将成为手机等无线互联网的老大。

随着 Gears 的推出及发展，正如本节前面所说“Gears 是互联网向桌面融合的产物”。互联网向桌面融合，其实就是网络操作系统概念的一部分。从长远来看，好比当 Adobe 和微软正在努力怎么多喝汤的时候，Google 却已经养肥了一条牛，正准备喝血吃肉。

优劣到底如何？现在很多人没法远看，即使能真正看清眼前实况的人也特别少，绝大多数人只是在听那些一知半解的“砖家”的长篇大论，甚至在听谬论，以至于自己越听越糊涂。

我不是专家，更不是“砖家”，但我通过我的文字来表达我的思想，争取让不懂 IT 的人也能看懂我的思想。

我的思想是：① IT 需要简单化，更需要系统化；② 人生需要引导，编程更需要引导；③ 编程的核心是思想，语言只是思想的一种实现方式而已，当你有了思想，即使不懂某种语言，也很快就能用其实现功能。

我们再看看和 Gears 相关的知识。

Gears 是一个 Open Source Project，参与者有 Adobe、Mozilla 等。没错，就有 Adobe，大家会觉得比较奇怪吧，原本 Adobe 一心想用 AIR 吃掉 Desktop 与 Off-line Market，怎么还跑去跟 Google 合作呢？

其实一点也不奇怪，正如前面所说：“AIR 只是一个和 Ajax 组件包一个层次的解决方案，主要用于客户端表现。AIR 技术的本质就是 HTML + JavaScript”。也就是说，AIR 没法成为 Gears 真正的竞争对手，反倒是 AIR 可以借用 Gears 的部分功能，如通过 HTML Wrapper 里的 JavaScript 去呼叫 Gears，用 Datastore 存取 DB 与 Local server，这样 App 还可用 swf 来做，但 local persistence 的事就可交给 Gears。必要时，AIR 还可以用 Gears 来 cache 一些东西，所以 Gears 对 AIR 有利无害。

常用的应用，如电话簿、CD/相片管理程序、进销存系统、ERP 等，每个产品都需要一个 local storage(本地存储)。如果 AIR 的目标是进军 Desktop，即要打破 Browser/Desktop RIA 的界线，那么 AIR 自备一个可用的 Data Storage 就是最基本的功能。既然 Gears 已经有了这个功能，那么 AIR 拿来使用也就不足为怪。

11.9 通过 RIA 看互联网本质

网络时代的竞争向来是赢者通吃的惨烈博弈。占据第一的位置意味着：广泛的用户、更多开发者支持、有资格成为某种行业标准的制定者以及股东腰包的鼓胀度。简单归纳为两个字——生存。不做第一，很难生存。

网络服务已成大势。这意味着往后用户的计算机都将成为网络终端，所有的资料都将在网络存储，所有的软件都基于浏览器服务。这不是妄想，Gmail、Google Notebook、Gears 等的出现已经告诉我们，这个时代很快就要到来。

有人说，虽然微软在网络服务方面很一般，但 Ajax 和 Flash 的命根子却都被微软捏在手里，只不过稍稍顾及反托拉斯竞争法不敢轻举妄动而已。不妄动归不妄动，小动作那可从来没停过，如稍稍改了下 ActiveX 的工作方式，就让整个 Flash 世界忙活了个底朝天。这个我没经历过，因为我的浏览器都是禁止 Flash 的。有人提到，如果微软哪天再改改 JavaScript 虚拟机的运行方式，或者改改 IE 对 JavaScript 的运作方式，那又轮到 Google 忙活了。

真会这样吗，微软敢这样吗？尽管微软手拥尚方宝剑——操作系统，这在网络服务还

没成型时微软确实是老大，但如今已进入网络时代，网络大佬们早就思考过微软的可能策略及相应对策。微软如使用非常手段，只会众叛亲离，更多公司将投入对手怀抱。何况，微软在大型企业应用中根本无能为力，只能在中小型企业玩玩。随着互联网的发展，中小型企业的要求也越来越高，企业信息安全、有线及无线网络同样无缝使用、各种终端的适应等问题，基于微软的构架起码在目前很难适应，至于以后能否适应，只有微软自己才知道。

RIA 实现了比传统 HTML 网页更强力、反应更迅速、界面更华丽的 Web 表现。对于用户来说是一次用户体验上的革命，无论是视觉上，还是操作习惯上，都会发生较大的变化。目前成熟的应用，如在线地图，拥有非常直观的图形界面和操作，并且能立即与网络服务器交换信息。整个富客户端的改变，最终会改变用户的上网习惯。在摆脱了浏览器的约束以后，RIA 的应用范围更大地被扩展了，不过 RIA 也只是一种表现方式而已。

相比传统的 SWF 应用，AIR 在技术上加入了文件系统的 I/O 操作 API 和 HTML 渲染引擎(其中包含 JavaScript 解释器)。就像 Ajax 根本不能称之为一种“技术”一样，AIR 仍然只能被理解为 XHTML + CSS + ActionScript + JavaScript。但 AIR 最大的特点及最引人注目的地方在于，“AIR 能构建便携式的无实时连接的 Web 程序”。Google Gears 的出现及壮大，证明了这种模式的必然性。

互联网软件在向客户端融合，客户端在向互联网融合，无线在和有线融合，这是互联网应用的实质。我们再来看 Adobe 副总裁托马斯 • 黑尔对 AIR 的解释，“如果你关闭了浏览器，谷歌地图就无法再与你‘交谈’了” 。如果计算机也关闭了呢，AIR 怎么处理？有了无线和有线的融合，服务可以无处不在，这也是 Google 在互联网做大后，就推出手机操作系统的一个原因。

互联网企业发源于 Web 世界，所以要延伸互联网，就必须基于现有优势。JS 技术，这种根植于网络世界的技术，就是最理想的选择。习惯使用 Web 应用软件的用户，对于本地安装软件、本地软件那样的操作习惯会感到很奇怪。而对于习惯使用本地软件的用户，当有跨出局域网的业务需求时，B/S 企业管理软件应运而生。

在企业管理软件的开发上，既开发 C/S，又开发 B/S，极其耗费时间和成本，而以后的维护更耗成本。所以，还不如做一个既是 C/S，又是 B/S 的软件。

Java、PHP、C#等语言能够处理任意的 HTML 网页，再通过数据库实现大部分的存储操作。但这是仅能部署在服务器端的一种应用，对于操作浏览器等客户端方面常常力不从心。RIA 就是这样一个更强的前端表现层，但 RIA 分成了微软和 Adobe 两个阵营，并且水火不容，而且还有 Google 的 Gears 在捣局。

不少人向来对微软的东西不太感冒，因为除了臃肿外还有缓慢。尽管 C#属于 ECMA 标准语言系列，上手不难，框架也完整，但是和它配合的数据库是 MS SQL Server，换句话说，还是离不开 Windows 平台。尽管比尔大叔也有跨平台的野心，不过成熟度有限。Adobe 公司的 AIR 1.5.1 安装程序为 15MB，.NET Framework 4.0 Beta 1(x86)为 78MB，Silverlight 光实现 C#的.NET 框架和 MS SQL Server 这堆东西就已经有几百 MB 了。(题外

话，要查相关的中文资料，Adobe 一查就有，微软的就差多了。)

有人说微软已经在向广告型企业转型，但因为桌面应用这块肥肉实在太大而无法抛弃。不过微软的优势还是有很多，Windows 平台的使用者极多，霸主和垄断地位就是最大的武器。

Adobe 阵营的优势是 Flash 的广泛和成熟，Flash 支持 Socket，也能通过 AMFPHP 与 PHP 服务端进行通信，只需再掌握 Flash ActionScript 3 即可。跨平台方面，AIR 以 Java 为基层，在 Linux 上畅通无阻。Adobe 还可以通过部署在服务器的 Flex 实现更强的后端功能。

Google 是目前互联网的霸主，Gears 是为自家的 App 铺路，目前已经推出了一系列产品的离线版，它们都基于 Gears，而且其他公司也已经推出不少基于 Gears 的产品。以下是一些常用的使用 Gears 的网站应用列表：

1) Google Reader：Google 提供的阅读器，2007 年 5 月集成了 Google Gears，可以让读者离线阅读最近的 2000 篇文章，还可以对文章标记或共享。

2) Remember the Milk：任务管理中的应用，在 Google Reader 集成后的第 6 天就实现了集成 Google Gears 的第 2 个离线应用。在这个应用中，你不仅仅可以获得列表，还能添加新的任务和注释，编辑现有的任务(完成状态、轻重缓急、标签、改变交货期以及核心内容)，使用自己的个人任务，创造新的智能化任务管理。

3) Zoho Writer：在线文字处理，最近编辑的一些文件可以离线查看和编辑。

4) PassPack：在线密码管理，离线版本可以将你的数据下载到本地，并在本地访问和管理，进行修改并保存之后，下一次上线会同步本地的离线账户信息。

5) MindMeister：在线思维脑图，离线版可以让用户离线创建思维脑图，所有的改变都会保存在本地，下一次上网后会进行同步。网址为http://www.mindmeister.com。

6) Buxfer：个人财务经理，离线应用会保存登录验证信息，可以在离线状态先方便地登录到你的财务系统中，你的重要和敏感的财务数据会安全地在你的控制之中。

7) Autodesk Labs Project Draw：在线创建图表，应用可以运行在离线模式，当重新连接后会进行文件同步。

8) Google Docs：在线字处理应用，用户可以离线查看和编辑文档。

9) Picasa Web Albums Mobile：照片共享应用，离线模式支持 Windows Mobile 6 的触摸装置，可以让用户离线查看相簿。

最后再提提 AIR 的内容：

1) AIR 是给那些习惯开发 EXE 程序的开发者准备的非微软技术的互联网解决方案。

2) AIR 工作在表现层。AIR 只是一种客户端表现方式，本质是 HTML+JavaScript。不要奢望它能做各种业务处理和运算。你的业务可以包装成 Web Service，也可以是 PHP、JSP 或 ASP.NET 页面，调用方式可以是 Web Service、HTTP Service 或 Remoting。这样业务层就可以和表现层 AIR 交换数据。

3) AIR IDE 只是 HTML + JavaScript 的设计和调试工具，也别奢望 AIR IDE 能像 Java 和 C#的 IDE 环境那样强大。过去怎么设计和调试 HTML + JavaScript，现在还是一样。

4) AIR 不是 Flash。它们是两种不同的应用目的和技术发展方向。不要用 Flash 开发企业管理软件，也不要用 AIR 开发多媒体。虽然能，但不擅长，就如同你拿 C 来开发 Web 应用一样。AIR 不需要运行在浏览器中，不需要使用 Flash 容器。AIR 有自己的容器和运行环境。AIR 和微软的 EXE 一样安装和执行。

5) AIR 和 WPF 都处于不成熟期，可以做做项目，但不要把宝都押在它们上面。

6) 不要在 Google Web OS 和客户端 OS 之间的选择上犹豫，如果你是网站开发人员，要安心研究 Google RIA 技术。

7) 如果没有开发过 MVC 架构，也没有用过 COM 多层开发，那么就不会明白业务层和表现层的分工和通信。如果你一直想弄清楚怎么在 AIR 中写 SQL 语句或 Java 语句，以及如何在 Servlet 中写 JavaScript 和 HTML 代码，那么你还处在浆糊阶段，建议多去学学软件的体系构架，给自己的思想升升级。不过只要你用心去看前面的相关章节，就会有一个完整的软件构架思想。

8) RIA 不是万能的，当你的 PC 都没有运行时，任何 RIA 技术都无济于事，唯有结合基于无线网络的手机应用等技术才能解决问题。这也是 Google 推出手机操作系统的一个原因，因为 Google 以后将在企业应用领域大展身手。

第 12 章

HwCMS 内容管理系统详解

HwCMS 曾作为 howwe.net 的运行程序，提供网站服务；现作为一个基于 MySQL 的 PHP 应用系统，阐述 PHP 的完整应用，使读者能尽快掌握 PHP 的实用编程技术。

学以致用是本书倡导的理念。我们为什么要学习 PHP？该学哪些内容？具体怎么学？我想这是很多人应该思考、却很少有人去思考的问题。大部分人因为没人引导，有些迷茫或很迷茫。我们回到“学以致用”四个字，字面意思很简单——“学”的最终目标是“用”。HwCMS 有什么用呢？

HwCMS 曾用于浩为公司网站的宣传，为宣传内容的展示提供基础。CMS 是 Content Management System 的缩写，中文为“内容管理系统”，可见 HwCMS 是一个内容管理系统。内容管理系统通常是指门户或商业网站的发布和管理系统。如果定义宽泛一点，个人网站也可归入其中，WiKi 也一样，博客也可归入。不管系统怎么命名，但其实质不变，只是一个基于数据库的应用而已，唯一不同的是数据表现的方式各异。

如何显示数据，这只能根据实际需要来决定，而实际需要就是“用”。不仅如此，面对大用户的高并发访问，如何才能满足，这需要对系统进行优化，优化也是“用”，但这是高层次的“用”，需要时间及经验的累积。不过，也不用泄气，“万丈高楼平地起”，对网站开发来说，“平地”就是掌握 PHP 的基本使用方法——如何调用数据库。

注意：

运行本章提到的连接前，必须先运行 30-php-Start.bat，以加载 PHP 运行环境。

12.1 HwCMS 简介

内容是任何类型的数字信息的结合体，可以是文本、图形图像、Web 页面、业务文

档、数据库表单、视频、声音、XML 文件，等等。可见内容是一个比数据、文档和信息更广的概念，是对各种结构化数据、非结构化文档及信息的聚合。管理就是施加在“内容”对象上的一系列处理过程：收集、存储、审批、整理、定位、转换、分发、搜索、分析，等等，目的是为了使“内容”能够在正确的时间、以正确的形式传递到正确的地点和人。

内容管理可以定义为：协助组织和个人，借助信息技术，实现内容的创建、存储、分享、应用、检索，并在企业个人、组织、业务、战略等诸方面产生价值。而内容管理系统就是能够支撑内容管理的一种工具或一套工具的软件系统。

HwCMS V1.0 只是一套构架简单的内容管理系统，用于演示基于 MySQL 的 PHP 应用，使用户能快速掌握并实现相应的数据库应用。随后的改版将加入更多人性化操作。以人为本、方便用户的使用，是浩为公司设计软件的基本原则。希望更多的人能遵从这个原则，其实这也只是“学以致用”的一个体现：即使你的软件设计得再好、再先进，而没有人用，或用户很难用，就没达到软件设计的最终目标。

HwCMS 没有使用框架，仅对代码按 MVC 进行了分层，但 HwCMS 没有实现真正的 MVC 构架，因为 PHP 根本就无法实现真正的 MVC 构架。其整体思路是：按照功能的需要，在 index.php 中调用功能函数或在 admin.php 中包含功能实现文件，然后再包含相应页面的 HTML 文件，从而实现相应的功能。即 index.php 及 admin.php 是 MVC 中的 C(**Control**-控制器)，功能函数及功能实现文件是 M(**Model**-数据模型)，而页面的 HTML 是 V(**View**-用户界面)。

在浏览器中运行 http://127.0.0.1/，首页部分截图如图 12-1 和图 12-2 所示。



图 12-1　HwCMS 首页部分截图一

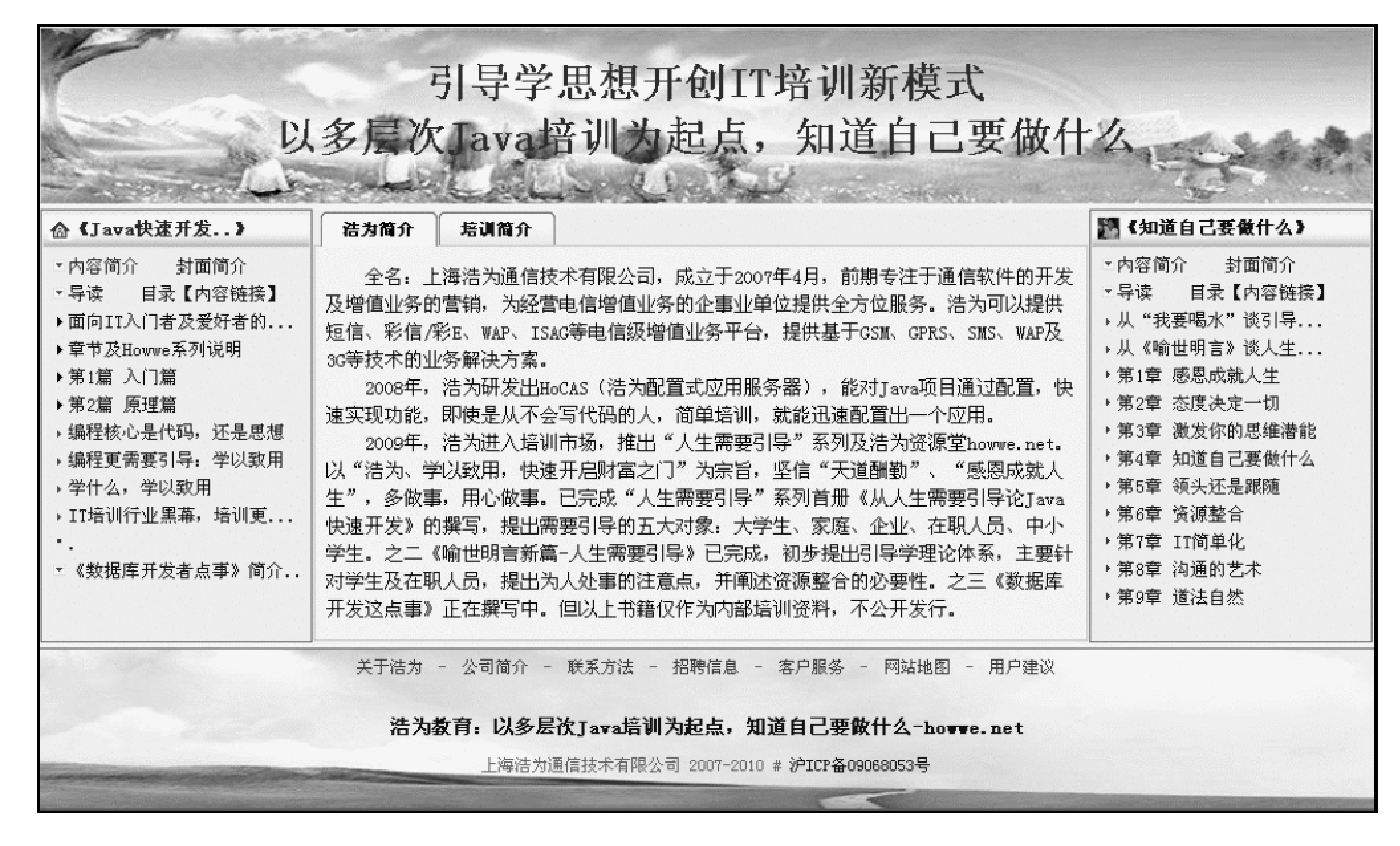


图 12-2 HwCMS 首页部分截图二

如图 12-1 与图 12-2 所示，这样就组成了一个像模像样的公司主页，基本展示了公司业务。注意，当时为了简单，仅手工编写了一个静态页面，在需要修改时可以手工调整内容，而真正的 CMS 首页是系统自动生成的静态页面。

为了让 index.html 成为默认首页，在 D:\howwe\php\Apache2\conf 的 httpd.conf 文件中有如下设置：

```
<IfModule dir_module>
  DirectoryIndex index.html index.htm index.php index.jsp
</IfModule>
```

注意：

依照首页文件列表，如果存在 index.html，则 index.html 为默认首页；否则查找 index.htm，如果存在，则 index.htm 为默认首页。其余的依此类推。

如果不能修改默认首页的文件列表，可在.htaccess 中添加相应内容，例如：

```
DirectoryIndex index.html index.htm index.php index.jsp.
```

运行 http://127.0.0.1/index.php，部分截图如图 12-3 所示。

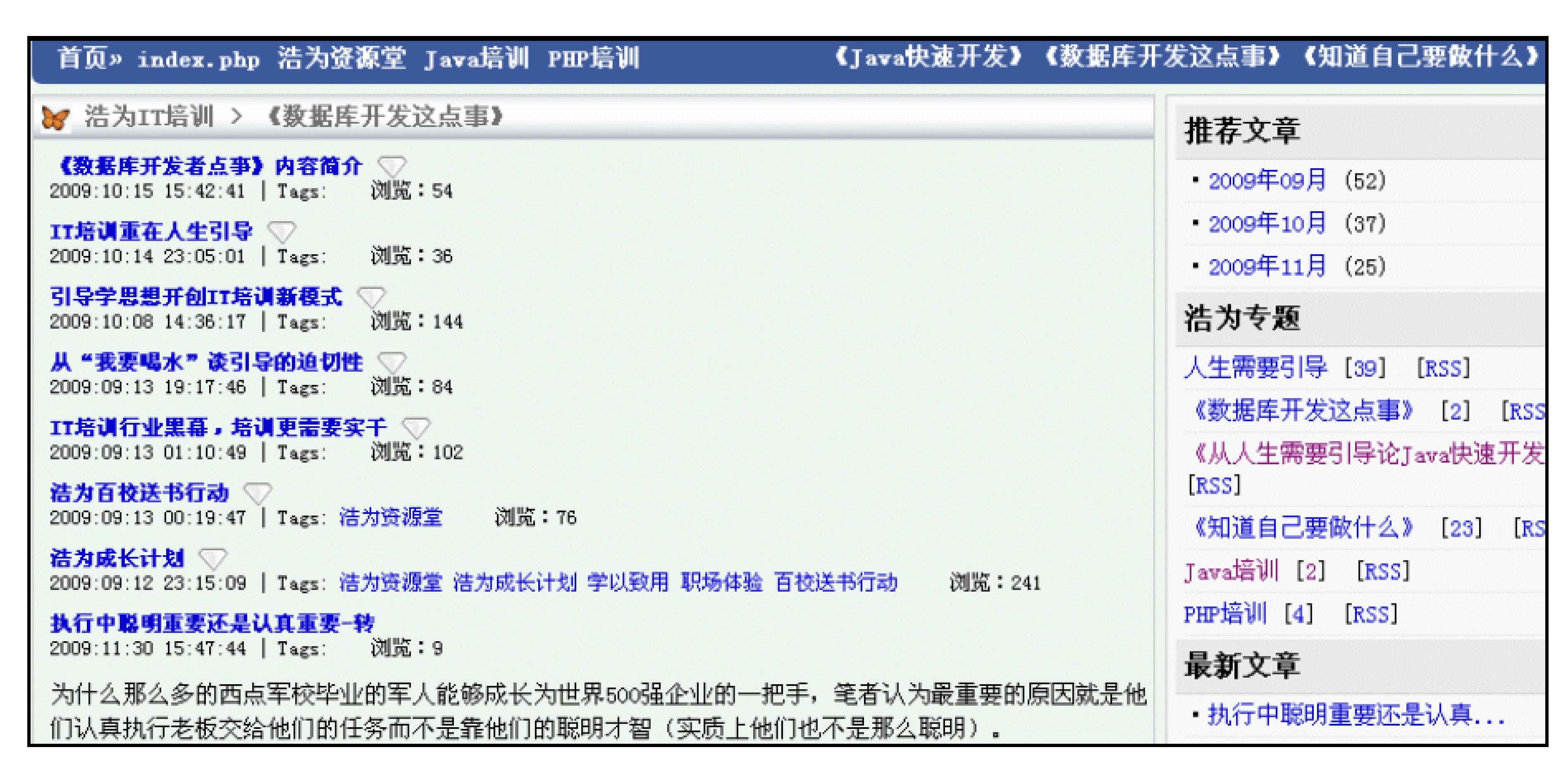


图 12-3　HwCMS 首页-index.php 部分截图

如图 12-3 所示，显示了文章列表及部分内容。

点击“引导学思想开创 IT 培训新模式”链接，页面截图如图 12-4 所示。



图 12-4　文章内容部分截图

运行 http://127.0.0.1/admin.php，出现登录窗口，用户名、密码均为 admin，管理后台截图如图 12-5 所示。

欢迎您 admin [注销身份] [后台首页] [站点首页]
首页 设置 分类 文章 HTML 评论 标签 引用 模板 数据 备份 用户 注销
管理首页
基本信息
程序版本：V1.0
服务器环境
操作系统及PHP：WINNT / PHP v5.2.8
mysql版本：5.1.40-community
上传许可：10M
当前数据库尺寸：540.38 KB
magic_quote_gpc：On
关于浩为 - 公司简介 - 联系方法 - 招聘信息 - 客户服务 - 网站地图 - 用户建议

图 12-5　管理后台截图

图 12-5 所示的管理后台在“**12.7 管理后台功能的实现中**”有详细说明。

按照以上顺序，读者应该对 HwCMS 的操作有了初步的印象，同时更能明白其作用。下面详细介绍其实现。

12.2 数据库范例

在“**7.2 数据操作种类**”中示范了一个调用数据库的例子 D:\howwe\wwwroot\0801.php，其实该例修改自 1301.php，文件位置为 D:\howwe\wwwroot\1301.php，我们先看下效果。运行 http://127.0.0.1/1301.php，截图如图 12-6 所示。

title	artid	catename	catedir
执行中聪明重要还是认真重要-转	142	人生需要引导	plan
管理的真谛：“轮流分粥，分者后取”	141	人生需要引导	plan
【浩为资源堂月刊】从网瘾谈引导的迫切性	140	《知道自己要做什么》	bk2
【浩为资源堂月刊】从“我要喝水”谈感恩教育	139	《知道自己要做什么》	bk2
【浩为资源堂月刊】小语句解决大难题，IT需要简单化	138	《从人生需要引导论Java快速开发》	bk1
【浩为资源堂月刊】脚踏实地做事	137	《知道自己要做什么》	bk2
【浩为资源堂月刊】真能持才傲物，仅是无知而已	136	《知道自己要做什么》	bk2
【浩为资源堂月刊】学会留下你的脚印	135	《知道自己要做什么》	bk2
【浩为资源堂月刊】成功者与普通人的差别：知道自己要做什么	134	人生需要引导	plan
【浩为资源堂月刊】优秀是种习惯	133	《知道自己要做什么》	bk2
【浩为资源堂月刊】活着是一种责任，多做善事	132	《知道自己要做什么》	bk2
【浩为资源堂月刊】我的成长过程：一个责任感增强的过程	131	人生需要引导	plan

图 12-6 数据库范例截图

再看代码：

```
<?php
    require_once 'inc/common.php';
    $artdb = array();

    $sql = "select id,title,cateid,catename,catedir,istop from hw_articles
            order by id desc";
    $result = $db->query($sql);
    while($article = $db->fetch_array($result)){
        $article['artid'] = $article['id'];
        if($article['istop'] == 1){
            $article['istop'] = '置顶';
        }else{
            $article['istop'] = '无';
        }
        $artdb[] = $article;
    }
?>

<table>
```

```
        <tr>
          <td>title</td>
            <td>artid</td>
            <td>catename</td>
            <td>catedir</td>
        </tr>

        <?php
        foreach($artdb as $art){
        ?>
        <tr>
          <td><a href="<?= $art['catedir']?>/<?= $art['artid']?>.html"><?=
                $art['title']?></a></td>
            <td><?= $art['artid']?></td>
            <td><?= $art['catename']?></td>
            <td><?= $art['catedir']?></td>
        </tr>
        <?php
        }
        ?>
    </table>
```

代码说明：

1) `require_once 'inc/common.php';`引入包含文件，而 common.php 设置了一些参数，并包含了很多文件：inc/config.php 和 inc/db.class.php 为数据库连接文件；inc/global.function.php、inc/blog.function.php、inc/template.php 与 inc/cache.php 为功能函数；cache 目录下的文件为缓存文件。

2) 数据库相关的代码：

```
$sql = "select id,title,cateid,catename,catedir,istop from hw_articles order
        by id desc";
$result = $db->query($sql);
while($article = $db->fetch_array($result)){
```

只有这三句，查询结果通过`$article`被读入`$artdb`。

3) `<table>`与`</table>`之间的代码用于显示数据，数据在`$artdb`中，所以通过 foreach 来读取。

总之，整个流程如下：先将数据库连接文件包含进来，然后定义查询语句，查询后再对查询结果进行处理，最后将数据显示出来。

12.3 数据库连接详解

数据库连接文件只有两个：config.php 和 db.class.php。文件位置为 D:\howwe\wwwroot\inc，需要访问数据库时，写上 require_once'inc/common.php'; 即可。

下面我们修改 1301.php 中的包含文件 require_once 'inc/common.php';，将其修改成两个单独的数据库连接文件，保存为 1302.php，运行结果与图 12-6 一样，修改代码如下：

```
require_once 'inc/config.php';
require_once 'inc/db.class.php';
```

下面分别说明这两个文件，config.php 其实只是定义了 5 个变量，代码如下：

```
<?php
// 数据库主机名或 IP
$dbhost = '127.0.0.1';
// 数据库用户名
$dbusername = 'root';
// 数据库密码
$dbpassword = '1';
//数据库名
$dbname = 'a0903171509';
// 数据库表前缀
$tablepre = 'hw_';
?>
```

db.class.php 代码如下：

```
<?php
class db{
    var $connstring;
    var $querynum = 0;
    function connect($dbhost,$dbusername,$dbpassword){
        $this->connstring=mysql_connect($dbhost,$dbusername,$dbpassword)
                              or die($this->halt('数据库连接失败！'));
        return $this->connstring;
    }
    function select_db($dbname){
        mysql_select_db($dbname) or die ($this->halt('数据库选择失败！'));
    }
    function query($querystring){
```

```
        $result = mysql_query($querystring,$this->connstring);
        if(!$result){
            $this->halt('查询失败! ',$querystring);
        }
        $this->querynum ++;
        return $result;
    }
    function fetch_array($result){
        $record = mysql_fetch_array($result);
        return $record;
    }
    function num_fields($result){
        $fields = mysql_num_fields($result);
        return $fields;
    }
    function num_rows($result){
        $rows = mysql_num_rows($result);
        return $rows;
    }
    function insert_id(){
        $insertid = mysql_insert_id();
        return $insertid;
    }
    function rows_count($sql){
        $result = $this->query($sql);
        $rowsnum = $this->num_rows($result);
        return $rowsnum;
    }
    function free_result($query){
        $query = mysql_free_result($query);
        return $query;
    }
    function getversion(){
        return mysql_get_server_info($this->connstring);
    }
    function fetch_all($sql, &$arr){
        $query = $this->query($sql);
        while($data = $this->fetch_array($query)) {
            $arr[] = $data;
        }
    }
    function halt($msg = '',$sql = '') {
        $output = '<div style="font-size:11px;font-family:Verdana">msg:
```

```
                    '.$msg.'<br />sql:'.$sql.'<br />errno:'.mysql_errno().
                    '<br />error:'.mysql_error().'</div>';
            exit($output);
        }
    }
    $db = new db();
    $db->connect($dbhost,$dbusername,$dbpassword);
    $db->select_db($dbname);
    mysql_query("SET NAMES 'utf8'");
    ?>
```

代码说明：先定义一个名为 db 的类(class)，db 中定义了常用的 12 个函数(function)；然后再定义一个变量$db（$db = new db();），连接并选择数据库，最后将其编码设为 utf8。因此，在需要访问数据库时，就可以通过$db 来操作了。

12.4　HwCMS 目录说明

HwCMS 目录及文件如图 12-7 所示。

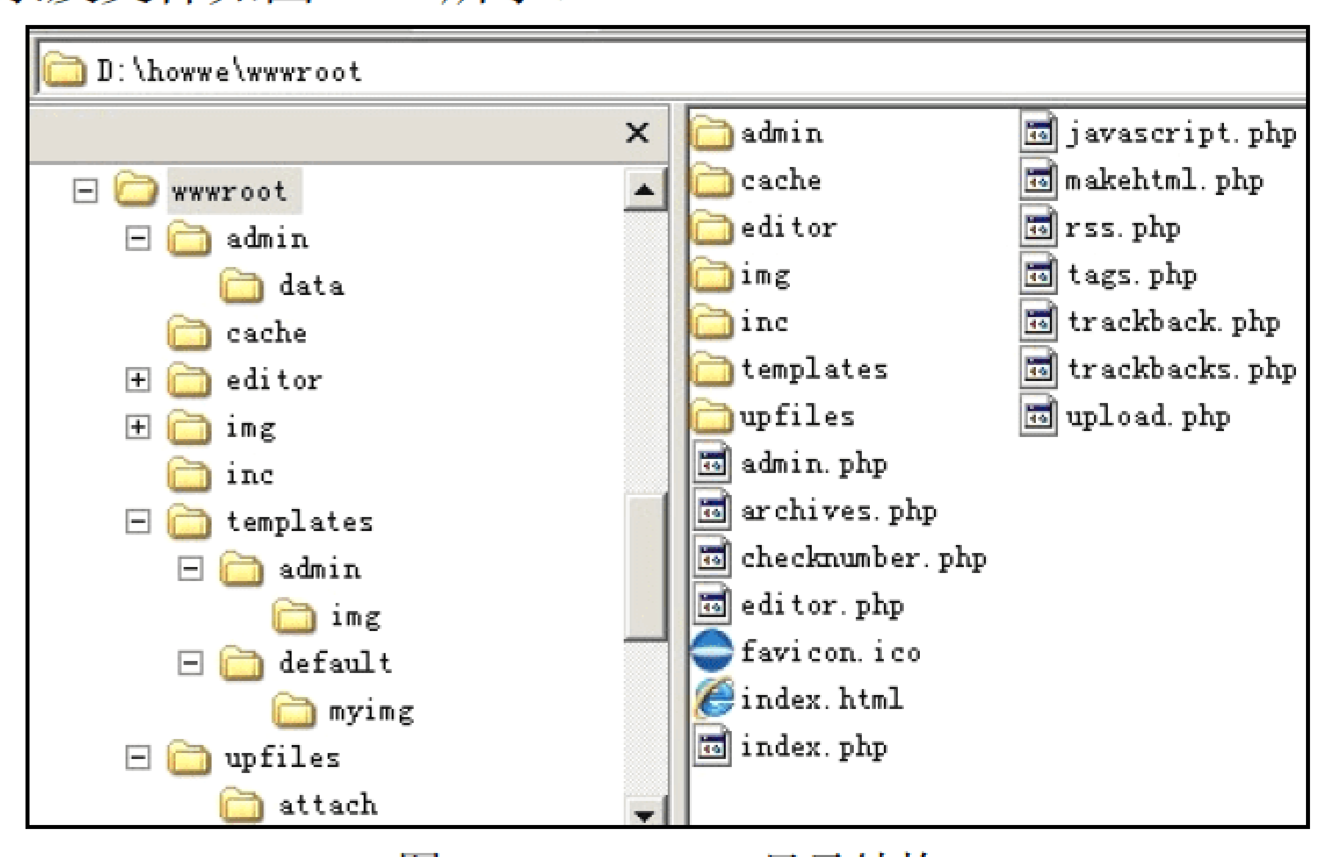


图 12-7　HwCMS 目录结构

注意：

img 目录只用于存放 p.css 及相关图片，非必需目录。

目录及重要文件说明：

- admin：存放管理后台的功能实现文件，有 articles.php、categories.php、comments.php、database.php、datas.php、html.php、index.php、login.php、setting.php、tags.php、templates.php、trackbacks.php、users.php，共 13 个，相应的 HTML 文件在目录

D:\howwe\wwwroot\templates\admin 下，具体说明请参考“**12.7 管理后台功能的实现**”。

- cache：缓存文件，基本上是根据数据库内容生成的数组，如 cache_archives.php、cache_calendar.php、cache_categories.php、cache_recentart.php、cache_recentcomment.php、cache_statistics.php。另外，cache_setting.php、cache_templates.php 中为基本配置信息。
- editor：编辑器 editor 源码。
- inc：功能文件，包含两个数据库连接文件、4 个功能实现文件及一个供引用的包含文件。
- templates：模板文件，由基本模板和后台管理模板构成。
- upfiles：存放上传文件。
- admin.php：管理后台主控文件。
- checknumber.php：生成图形码。
- editor.php：调用文本编辑器 editor。
- index.html：静态首页。
- index.php：动态首页及前台功能主控文件。
- javascript.php：Ajax 服务端代码。
- makehtml.php：生成静态文件。
- upload.php：文件上传处理代码。

说明：

index.php 与 admin.php 作为 MVC 结构中的 C(Control，控制层)，实现了将相应功能的文件动态包含进来。

12.5　MVC 及模板

由于设计模式的风行，MVC 随处可见，PHP 领域也不例外，虽然 MVC 结合模板能解决美工和程序的分离，即能使 PHP 代码和 HTML 代码分离，但 MVC 真的适合 PHP 吗，M、V、C 又该如何实现呢？

1. 数据与 MVC

我们先看看“**9.7 学什么：学以致用**”中对 MVC 的简单描述：无论是使用 JSP，还是使用 Struts，或是 SSH，我们都至少需要一些必需的元素，如果没有这些元素，或许我们真不知道程序能写成什么样子。

数据：在 Hello World 的例子中，就是 Cnt，它们共同构成了程序的核心载体。事实上，

我们往往会有一个 Hello 类来封装 Cnt，这样会使得我们的程序更加 OO(Object-Oriented，简称 OO)，即面向对象。无论怎么说，数据都会穿插在这个程序的各处，成为程序运行的核心。

页面展示：在页面上，我们需要利用 HTML，把我们需要展现的数据都呈现出来。同时我们还需要完成一定的页面逻辑，例如错误展示、分支判断，等等。

处理具体业务的场所：不同的框架，处理具体业务的场所不一样。原来可以用 JSP 和 Servlet，后来用 Struts 的 Action。

上面的这些必须出现的元素，尽管在不同的年代被赋予了不同的表现形式，但是拨开这些外在的表现形式，我们就可以发现，这不就是 MVC 吗？

数据 -> **Model**

页面展示 -> **View**

处理具体业务的场所 -> **Control**

图 12-8 所示是以 JSP 为例的模型图，这是一幅流行了很多年的讲述 MVC 模型的图。

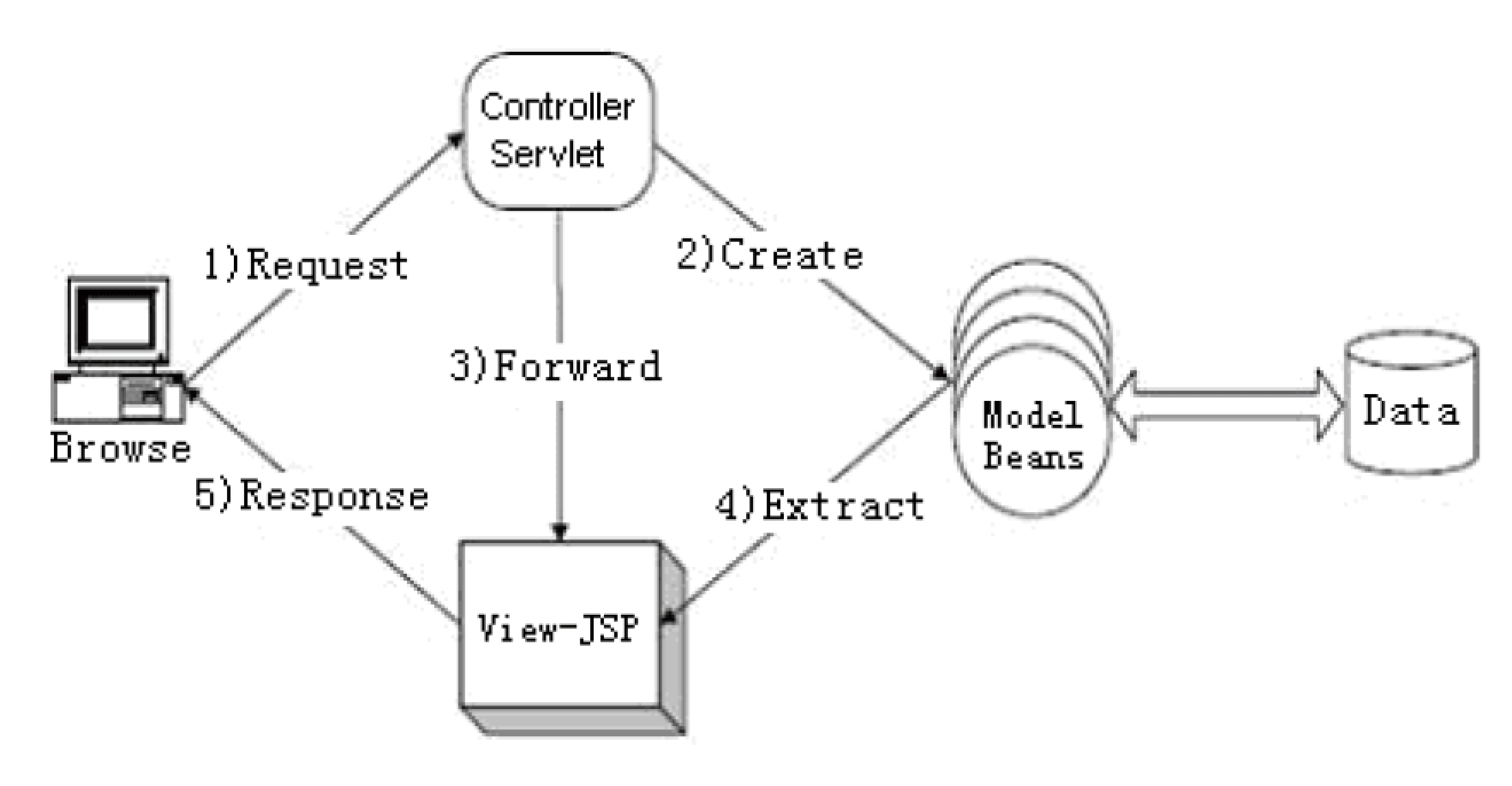


图 12-8 MVC 模型图

在这幅图中，MVC 三个框框各司其职，结构清晰明朗。但这幅图忽略了一个问题，数据是动的，数据在 View 和 Control 层一旦动起来，就必须进行很多必要的处理。

2. MVC 原型

MVC 本来存在于桌面程序中，M 为数据模型，V 为用户界面，C 则是控制器。使用 MVC 的目的是将 M 和 V 的实现代码分离，从而使同一个程序可以使用不同的表现形式。比如一批统计数据，你可以分别用柱状图、饼图来表示。C 存在的目的则是确保 M 和 V 的同步，一旦 M 改变，V 就应该同步更新。

3. Java 中的 MVC

Java 把 MVC 引入 Web 领域，并在此基础上设计出 Model2 体系。由于 Web 的特殊性，

Java 中的 MVC 和桌面程序中的 MVC 并不完全一致。主要原因是 Web 中的 V 不能持续，用户每访问一次，V 就重新生成一次，即 V 始终和 M 一致，不需要 C 来控制同步。那么 Java 中 C 的作用是什么呢？C 通常用于流程转向，它使用的是 Dispatch 模式，不再是桌面程序中的 C。

4. PHP 中的 MVC

要在 PHP 中原封不动地 Copy Java 的 MVC 是不可能的。主要表现在 M 上，在 Java 中，M 是独立于业务逻辑和表现逻辑的数据模型，能在服务器端跨页面存在，JavaBean 扮演这个角色。但 PHP 由于对象无法持久，即进程不能长时间驻留于内存，根本就无法直接实现 M。大多声称实现了 MVC 模式的 PHP 程式，都通过模拟手段来实现 M。实现的方式一般是在当前页面结束前把数据存入数据库或 cookie/session，在下一个页面中再通过数据库或 cookie/sessio 来重建 M。这样则开销巨大，本来往内存中写数据的简单操作，现在要从服务器端传到客户端或数据库，然后再传回来。与其用这么大的开销来维护一个数据模型，再在最后把这个模型塞回数据库，还不如直接根据需要更新数据库里的数据。可见，PHP 可用的 MVC 架构可以用图 12-9 表示。

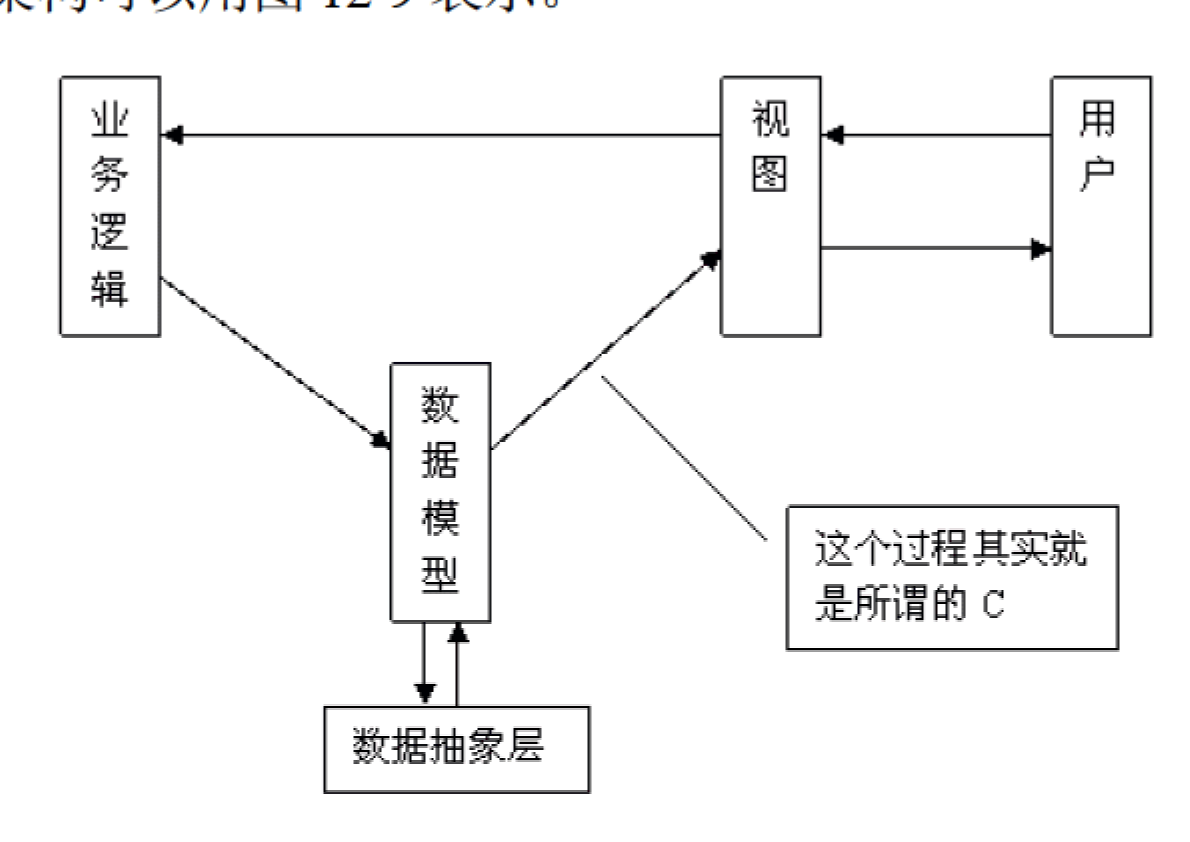


图 12-9　PHP 可用的 MVC 架构

5. 模板简介

我们先看模板引擎的实现原理。一般来说，有如下步骤：

(1) 注册变量，把从 Model 返回的数据注册到模板引擎中，告诉模板引擎这个变量可以使用。

(2) 模板解析，先读取模板文件，按照模板语法将标签解析成 PHP 语法，或者执行一些替换操作，用变量内容替换掉模板标签。其实效果都差不多。

(3) 如果不是将变量内容替换掉模板标签，那么基本上第三步就是将注册的变量和解析完的模板融合生成 PHP 文件，存放在专门的文件夹下以供下次调用。

当然，有些模板引擎还会提供不少用于显示内容处理的插件，如日期转换、字符串处理、生成表格、生成 select 等，这给页面制作提供了不少便利。有些还提供页面缓存机制等功能，这能提高 PHP 的性能。

但很多模板引擎都顶着语法简单的噱头，提供了一套专门的模板语法，我们先对比对比下面两种语法：

1) PHP 语法

```
<!-- <?php foreach ($lists as $value) { ?> -->
    <?=value['userName']?>
<!-- <?php } ?> -->
```

2) 相应模板的语法

```
<!-- loop lists value -->
    {value['userName']}
<!-- loop -->
```

大多模板的语法美其名曰可以降低美工的学习门槛，可是，有多少模板是由美工来完成的呢？何况对比两种语法，一般人不会觉得 PHP 的简单循环和输出难以理解。

不仅如此，模板文件最终将被替换成 PHP 文件，也就是说，不管哪种模板引擎语法，最终都会被转换成标准的 PHP 语法。而模板引擎的解析，也只是将模板语法转换成 PHP 语法的过程。但模板引擎既然存在，就有存在的理由，某些场合不得不用：例如把模板提供给用户去制作和使用，就得采用标签以限制用户使用 PHP 语法，从而增强系统的安全性。

最后，再说说效率问题。由于模板引擎要解析模板语法，会用到很多正则匹配和替换，这在实际运行中很消耗系统资源，而且当模板标签非常复杂或嵌套多层的时候，效率会很低。尽管可以通过预编译，把带有模板语法的模板转换成 PHP 语法的文件来直接包含，可编译后的结果文件比我们写出来的 PHP 文件复杂得多，效率远低于直接编写的 PHP 文件。有兴趣的读者可以去 Google 学习相应知识。

为了能使用户快速掌握相应技术，我们在 HwCMS 的讲解中，不使用模板语法，而直接使用 PHP 语法，以减少学习难度。

12.6 前台功能的实现

前台的功能不多，仅有图 12-3 和图 12-4 所示的页面，其中 index.php 用于对文章标题及部分内容按照不同条件进行显示。

1. index.php 的代码

```
<?php
$upcache = isset($_GET['upcache'])?$_GET['upcache']:'';
if ($upcache=='1'){
  $sinfo="Location:/";
  Header($sinfo);
}
require 'inc/common.php';
$blog_title = $options['blog_title'];
$keywords = $options['blog_keywords'];
$description = $options['blog_description'];

$calendar = calendar();
$action = isset($_GET['action'])?$_GET['action']:'';
switch($action){
    case '':
    $articledb = artlist();
    break;
    case 'cates':
    $articledb = artlistBycate();
    break;
    case 'times':
    $articledb = times();
    break;
    case 'tags':
    $articledb = tags();
    break;
    case 'search':
    $articledb = search();
    break;
}
categories_recache();
statistics_recache();
recentart_recache();
require template($templatefile,$templatename);
?>
```

代码说明：index.php 作为前台 MVC 结构中的 C(控制层)，实现了按照显示的需要来获取相应的文章列表。

1) 先获取`$upcache`的值，如果值为 1，则跳转到根目录。注意，该跳转判断是为了整合其他程序而设的，和 HwCMS 无关。

2) require 'inc/common.php';可引入包含文件。

3) 读取$blog_title、$keywords、$description的值。

4) 定义$calendar。

5) 读取$action，根据它的值通过 switch 判断并进而通过功能函数读取相应的文章列表信息。

6) categories_recache()、statistics_recache()和 recentart_recache()用来重建相应的缓存文件。

7) 包含 normal.php 或 list.php 是为了显示文章列表信息，具体文件由 template()函数返回。

需要注意的是：calendar()、artlist()等函数为功能函数，定义在 blog.function.php 中，简单看看就能明白。

2. 模板文件

正如“**12.5　MVC 及模板**”所述，为了使读者能尽快入门，HwCMS 把 MVC 结构中的 M 和 V 进行分离，同时也没有采用复杂多样的模板语法。

通过 index.php 中的 require template($templatefile,$templatename);可引入模板文件，其实 template()函数只返回需要包含的具体文件地址，而没有采用真正的模板技术。本书的目的是“学以致用”，因而先讲述一些能马上就能见效的实用技术，以增强读者自己的信心。模板技术请参考“**第 13 章　浩为资源堂代码修改详解**”或自己去 Google。

index.php 引入的模板文件为 normal.php 或 list.php，两个文件的格式基本一致，但前者比后者多显示了文章的摘要，对应效果如图 12-10 和图 12-11 所示。



图 12-10　HwCMS 首页-index.php(显示摘要)



图 12-11　HwCMS 首页-index.php(无摘要)

list.php 的代码如下(文件在目录 D:\howwe\wwwroot\templates\default 下)：

```
<?php
require 'header.php';
?>
<div id="container_body">
 <div id="main">
  <div id="cstitle"><img src='/img/fm.gif'><? $cateInfo = $articledb[0] ?>
   <span class="crumb cDGray"><a href="http://howwe.net">浩为 IT 培训</a> &gt;
      <a href="/<?=$cateInfo['catedir']?>"><?=$cateInfo['catename']?></a>
      </span>
  </div>
 <?
  foreach($articledb as $article){
  if ($article['istop'] == 1){
 ?>
 <div class="post">
  <div class="title"><a href="<?=$article['pageurl']?>"><?=$article
      ['title']?>  <img src="/img/d.gif" alt="顶"/></a></div>
  <div class="info"><span><?=$article['updatetime']?></span>
   <span> | Tags: <? foreach($article['tags'] as $tag){?><a href=
        "<?=$options['url']?>/index.php?action=tags&tagname=<?=urlencode(
        $tag)?>"><?=$tag?></a> <? }?>   浏览:
        <?=$article['viewnum']?></span>
  </div>
 </div>
 <?
 }elseif($article['istop'] == 0){
```

```
      ?>
      <div class="post">
       <div class="title">
            <a href="<?=$article['pageurl']?>"><?=$article['title']?></a>
       </div>
       <div class="info"><span><?=$article['updatetime']?></span>
        <span> | Tags: <? foreach($article['tags'] as $tag){?><a
           href="<?=$options['url']?>/index.php?action=tags&tagname=<?=urlenc
           ode($tag)?>"><?=$tag?></a> <? }?>    浏览:
           <?=$article['viewnum']?></span>
       </div>
      </div>
       <?
       }}
       ?>
       <div class="fixed"></div>
      </div>
     <?php
     require 'siderbar.php';
     ?>
      <div class="bk6"></div>
      <div id="container_bottom">
       <div class="postnav">
        <div id="pageview">
         <ul>
         <? if($multipage != '')?>
          <?=$multipage?>
         </ul>
        </div>
       </div>
      </div>
     </div>
     <?php
     require 'footer.php';
     ?>
```

代码说明：

1) require 'header.php';用来引入头文件，菜单栏是头文件的一部分，具体代码请浏览同一目录下的 header.php，其效果如图 12-12 所示。



图 12-12 HwCMS 前台头文件

2) 在 id 为 container_body 的层(div)显示文章列表，在 id 为 cstitle 的层显示文章列表头，再遍历(foreach)文章列表变量$articledb，显示文件列表。

3) require 'siderbar.php';用来引入右侧的列表，其内容为“推荐文章”，先按月份显示每月的文章总数，再显示“浩为专题”和“最新文章”。

4) 如果文章需要分页，则在 id 为 container_bottom 的层(div)显示分页信息，如图 12-13 所示。



图 12-13 分页信息及页脚 footer.php

5) require 'footer.php';用来引入页脚。

normal.php 相比 list.php 能多显示文章的摘要，代码如下：

```
<div class="content">
   <?=$article['contentshow']?>
 <div class="fixed"></div>
</div>
```

3. 前台其他页面

1) 显示文章内容，模板页面为 show.php，在目录 D:\howwe\wwwroot\templates\default 下。页面代码格式类似于 list.php，由 makehtml.php 调用而生成静态页面，存放在相应分类所在的目录下，代码介绍请参考 12.7 节。

2) archives.php 按年和月生成如图 12-14 所示的文章总数列表，页面代码格式同 list.php，请自己阅读。代码说明略。



图 12-14 archives.php 显示文章总数

3) rss.php 生成 RSS 格式的内容，以供其他程序采集，显示内容如图 12-15 所示。

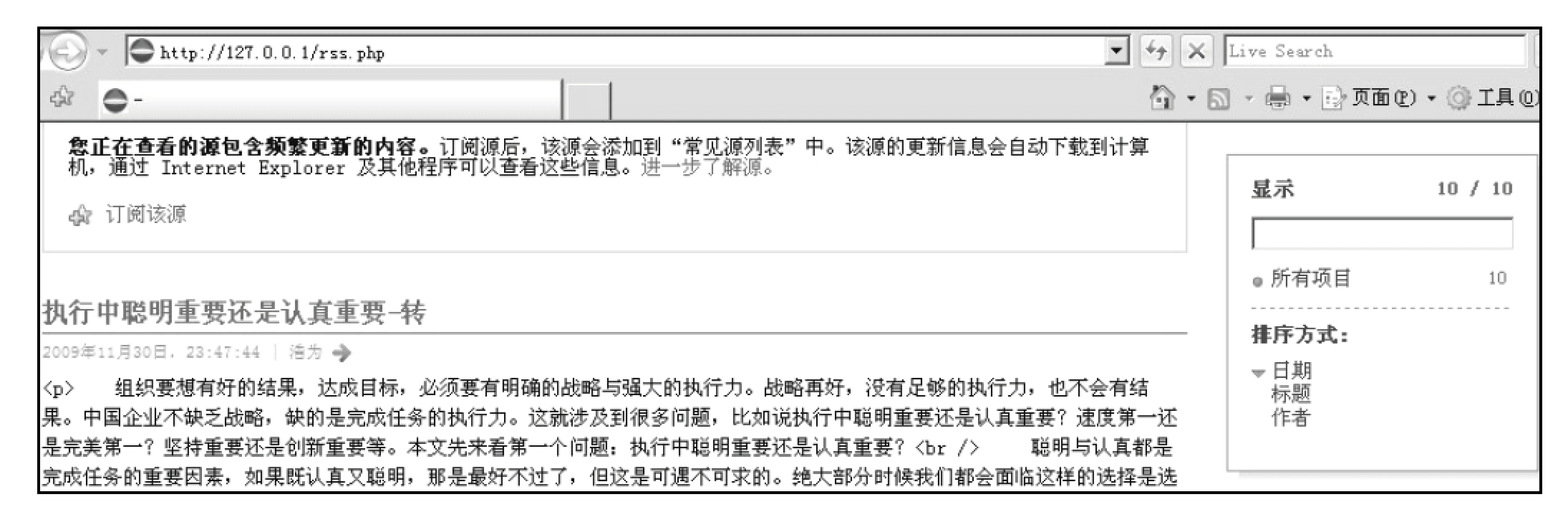


图 12-15 rss.php 生成 XML 文件

4) tags.php 显示标签，页面代码格式同 list.php，请自己阅读。代码说明略。显示内容如图 12-16 所示。



图 12-16 tags.php 显示标签

12.7 管理后台功能的实现

对一般用户来说，只能看到前台。其实，一个系统如果没有管理后台，运行参数就没地方可配置，自然系统也就没法运行，可见管理后台的重要性。

运行 http://127.0.0.1/admin.php，出现登录窗口，如图 12-17 所示。Username(用户名)、Password(密码)均为 admin，Checknumber 栏由 checknumber.php?a=shownumber 产生，该窗口由 D:\howwe\wwwroot\templates\admin\login.php 生成。登录成功后，管理后台的截图如图 12-18 所示。

图 12-17　登录窗口

图 12-18　后台管理界面

先依次单击菜单栏(图 12-18 中“首页”所在的横条)的各个菜单，再看看 admin.php 的代码，文件在目录 D:\howwe\wwwroot 下：

```
<?php
require_once 'inc/common.php';
$m = isset($_GET['m'])?$_GET['m']:'login';
$a = isset($_GET['a'])?$_GET['a']:'';
$menu = array(
    '0' => array(
        'm' => 'index',
        'a' => '',
        't' => '首页',
    ),
    '1' => array(
        'm' => 'setting',
        'a' => '',
        't' => '设置',
    ),
    '2' => array(
        'm' => 'categories',
        'a' => 'catelist',
```

```
            't' => '分类',
        ),
        '3' => array(
            'm' => 'articles',
            'a' => 'artlist',
            't' => '文章',
        ),
        '4' => array(
            'm' => 'html',
            'a' => 'article',
            't' => 'HTML',
        ),
        '5' => array(
            'm' => 'comments',
            'a' => 'commentlist',
            't' => '评论',
        ),
        '6' => array(
            'm' => 'tags',
            'a' => 'tagslist',
            't' => '标签',
        ),
        '7' => array(
            'm' => 'trackbacks',
            'a' => 'tblist',
            't' => '引用',
        ),
        '8' => array(
            'm' => 'templates',
            'a' => '',
            't' => '模板',
        ),
        '9' => array(
            'm' => 'datas',
            'a' => '',
            't' => '数据',
        ),
        '10' => array(
            'm' => 'database',
            'a' => 'tablelist',
            't' => '备份',
        ),
        '11' => array(
```

```
            'm' => 'users',
            'a' => 'userlist',
            't' => '用户',
        ),
        '12' => array(
            'm' => 'login',
            'a' => 'logout',
            't' => '注销',
        ),
    );
    if(in array($m,
       array('login','index','setting','categories','articles','html','comm
       ents','tags','trackbacks','templates','datas','database','users'))){
        if($m != ''){
            require 'admin/'.$m.'.php';
        }
    }else{
        exit('非法引用路径! ');
    }
    list($userid, $checknumber,) = $_COOKIE['admin'] ? explode("\t",
          authcode($_COOKIE['admin'], 'DECODE')) : array('', '');
    $userid = intval($userid);
    $result = $db->query("select username from ".$tablepre."users where id =
              ".$userid);
    $user =$db->fetch_array($result);
    $username = $user['username'];

    $filename = $m;
    require template($filename,'admin');
    ?>
```

代码说明：

1) require_once 'inc/common.php';用来引入包含文件。

2) 读取$m 和$a 的值。

3) 定义菜单栏内容变量$menu，共 13 项，再由 D:\howwe\wwwroot\templates\admin 下的 header.php 生成菜单栏，header.php 生成的页面内容如图 12-19 所示。



图 12-19　header.php 页面内容

注意：

header.php 不能直接通过 http://127.0.0.1/templates/admin/header.php 访问，因为没有定

义$menu，具体实现请自己看代码。

4) 检查 $m 的值是否在许可的功能列表中，如果在，就通过 require 'admin/'.$m.'.php';包含对应的功能实现文件，否则就终止。

注意：

功能实现文件中除 login.php 之外，其他都用 checklogin();来检查登录是否有效。

5) 通过$userid 获取$username。

6) 对$filename 赋值，并通过 require template($filename,'admin');包含相应的 HTML 文件。

例如：运行 http://127.0.0.1/admin.php 时，$m 为“login”、$a 为空，先包含功能文件 admin/login.php，再包含 HTML 文件 templates/admin/login.php，最后显示的页面如图 12-17 所示。提交之后，因为$a == 'login'，所以检查用户信息，验证正确则在显示提示信息后，再跳转到页面 admin.php?m=index，否则继续显示登录窗口。

在详细介绍几个重要的菜单项之前，先看看 HwCMS 的数据库结构。

1. HwCMS 的数据库结构(参见图 12-20)

Name	Type	Records	Size
hw_articles	MyISAM	114	526 KB
hw_categories	MyISAM	7	2,597 B
hw_comments	MyISAM	0	1,025 B
hw_sessions	MyISAM	2	2,081 B
hw_statistics	MyISAM	0	1,025 B
hw_tags	MyISAM	2	2,169 B
hw_trackbacks	MyISAM	0	1,025 B
hw_users	MyISAM	1	4,624 B

图 12-20　HwCMS 数据库的表信息

截图采用 D:\howwe\mysql\sqlfont 下的 SQL-Front.exe 作为数据库管理软件，共 8 个表，依使用的顺序分别为：hw_users(用户信息)、hw_sessions(登录信息)、hw_categories(分类信息)、hw_articles(文章)、hw_tags(标签)、hw_comments(评论)、hw_trackbacks(引用)、hw_statistics(统计)。后 3 个表没有数据，故没有列它们的表结构。前 5 个表详细情况如下。

1) hw_users(用户信息，参见图 12-21)

Name	Type	NULL	Default	Extras
Primary Index	id			unique
id	mediumint(8)	No	<auto_increme...	
username	char(20)	No		
password	char(50)	No		
email	char(50)	No		
homepage	char(50)	No		
flag	tinyint(1)	No	0	

图 12-21　用户信息表

存放用户的名称、登录密码、邮件等信息。

2) hw_sessions(登录信息，参见图 12-22)

Name	Type	NULL	Default	Extras
Primary Index	id			unique
id	mediumint(8)	No	<auto_increme...	
userid	int(11)	No		
checknumber	int(11)	No		
lastvisittime	int(10) unsigned	No	0	

图 12-22　登录信息表

存放登录用户的信息：用户编号、校验码(check number)、最后访问时间等。可根据最后访问时间对登录用户进行清理，即超过指定时间而无操作的用户将从该表中删除。

3) hw_categories(分类信息，参见图 12-23)

Name	Type	NULL	Default	Extras
Primary Index	id			unique
id	smallint(6) unsi...	No	<auto_increme...	
catename	char(50)	No		
catedir	char(20)	No		
pid	smallint(6)	No	0	
showno	smallint(6)	No	0	
flag	tinyint(1)	No	0	
keyword	varchar(50)	No		

图 12-23　分类信息表

根据 pid 和 id 可实现多级分类，子分类的 pid 为父分类的 id，顶级分类的 pid 为 0。

4) hw_articles(文章，参见图 12-24)

Name	Type	NULL	Default	Extras
Primary Index	id			unique
id	mediumint(8)	No	<auto_increme...	
istop	tinyint(1)	No	0	
cateid	smallint(6) unsi...	No	0	
catename	char(50)	No		
catedir	char(20)	No		
title	varchar(120)	No		
htmlurl	varchar(120)	No		
abstract	text	No		
content	mediumtext	No		
tags	char(50)	No		
commentnum	int(11)	No	0	
tbnum	int(11)	No	0	
viewnum	int(11)	No	0	
updatetime	int(10) unsigned	No	0	

图 12-24　文章表

5) hw_tags(标签，参见图 12-25 和图 12-26)

Name	Type	NULL	Default	Extras
Primary Index	id			unique
id	int(11)	No	<auto_increme...	
tag	varchar(120)	No		
artid	int(11)	No		

图 12-25　标签表

id	tag	artid
1	浩为资源堂 浩为成	1
2	浩为资源堂	3

图 12-26　标签表的内容

标签相关页面在前台的访问地址为 http://127.0.0.1/tags.php，文件在目录 D:\howwe\wwwroot 下，对应的 HTML 文件为 tags.php，文件在目录 D:\howwe\wwwroot\templates\default 下；后台管理地址为 http://127.0.0.1/admin.php?m=tags&a=tagslist，功能实现文件为 D:\howwe\wwwroot\admin\tags.php，相应的 HTML 文件为 D:\howwe\wwwroot\templates\admin\tags.php。多看看代码，很快就能明白。

2. 首页

首页是最简单的菜单项，如图 12-27 所示。在介绍其他菜单项之前，先介绍最简单的方式。



图 12-27　“首页”菜单项

“首页”菜单项的访问地址为 http://127.0.0.1/admin.php?m=index，可见其功能实现文件为 D:\howwe\wwwroot\admin\index.php，该文件先执行 `checklogin();`，检查是否已登录，然后定义了两个函数 `sizecount($filesize)` 和 `getserverinfo()`，最后将后一函数的执行结果赋值给 `$server`，供 HTML 文件调用，具体代码请自己阅读。

HTML 文件为 D:\howwe\wwwroot\templates\admin\index.php，代码如下：

```
<?
require 'header.php';
?>
<div id="submenu">
<ul>
<li>管理首页</li>
```

```
</ul></div>
<div id="cmain">
<table width="100%" align="center" cellspacing="0" cellpadding="5">
  <tr><td> </td></tr>
  <tr><td class="tbtitle"><h2>基本信息</h2></td></tr>
  <tr><td>程序版本：V1.0</td></tr>
  <tr><td  class="tbtitle"><h2>服务器环境</h2></td></tr>
  <tr><td>操作系统及 PHP：<?=$server['serverinfo']?></td></tr>
  <tr><td>mysql 版本：<?=$server['mysqlversion']?></td></tr>
  <tr><td>上传许可：<?=$server['fileupload']?></td></tr>
  <tr><td>当前数据库尺寸：<?=$server['dbsize']?></td></tr>
  <tr><td>magic_quote_gpc：<?=$server['magic_quote_gpc']?></td></tr>
</table>
</div>
<? require 'footer.php'?>
```

代码说明：

1) require 'header.php';用来引入头文件。

2) 在 id 为 submenu 的层(div)显示 6 个相应信息。

3) require 'footer.php'用来引入页脚文件。

提示：

在明白原理后，再看代码会很简单；否则就可能无从下手。

3. 分类

该菜单项用于管理 hw_categories(分类信息表)，内容如图 12-28 所示。

id	catename	catedir	pid	showno	flag	keyword
1	人生需要引导	plan	0	-4	1	
2	《从人生需要引导	bk1	0	0	1	
3	《知道自己要做什	bk2	0	0	1	
8	Java培训	java	0	0	1	
19	PHP培训	php	0	0	1	
23	帮助	help	0	0	0	测试修改关键词
26	《数据库开发这点	bk3	0	-1	1	

图 12-28　分类信息表的内容

再看对应的管理界面，如图 12-29 所示。

首页 设置 分类 文章 HTML 评论 标签 引用 模板 数据 备份 用户 注销

添加分类 管理分类

ID	分类名称	分类目录	父ID	显示	显示顺序	关键词	操作
1	人生需要引导	plan	0	☑	-4		编辑 删除 添加子分类
2	《从人生需要引导论Java快速开发》	bk1	0	☑	0		编辑 删除 添加子分类
3	《知道自己要做什么》	bk2	0	☑	0		编辑 删除 添加子分类
8	Java培训	java	0	☑	0		编辑 删除 添加子分类
26	《数据库开发这点事》	bk3	0	☑	-1		编辑 删除 添加子分类
19	PHP培训	php	0	☑	0		编辑 删除 添加子分类
23	帮助	help	0	☐	0	测试修改关键词	编辑 删除 添加子分类

图 12-29 “分类”操作界面

“分类”菜单项的访问地址为 http://127.0.0.1/admin.php?m=categories&a=catelist，可见其功能实现文件为 D:\howwe\wwwroot\admin\categories.php，该文件先执行 checklogin();，检查是否已登录，再根据$a 的值进行处理，一共 7 种情况：catelist、cateedit、catesave、childCateAdd、childCateAddSave、cateeditsave、catedel。具体代码请自己阅读。

HTML 文件为 D:\howwe\wwwroot\templates\admin\categories.php，代码格式与“首页”的一样，但会根据$a 的值显示不同的 HTML，共 4 种情况：catelist、cateadd、childCateAdd 和 cateedit。在代码中已用空行隔开，以便阅读。具体情况请自己阅读。

注意：

在编辑或添加分类时，只能处理主要信息。如果想处理所有信息，请自己修改。

唯一要讲解的是：页面中使用了 Ajax，如图 12-29 所示的复选框，通过 onclick="changFlag(this);"实现了动态更新。实现代码如下：

```
<script language="JavaScript" type="text/javascript"
     src="/img/js/ajax.js"></script>
<script language="JavaScript" type="text/javascript">
    function changFlag(field){
        //如果原来是选中(value=1)，则修改为不选中(value=0)
        if(field.checked){
            field.value = 1;
        } else{
            field.value = 0;
        }
        updateCateFlag(field.cateId,field.value);
    }
</script>
```

代码说明：

onclick 调用的 changFlag(this)；根据复选框的值，执行 JS 函数 updateCateFlag(field.cateId,field.value);。而该函数定义在 img/js/ajax.js 中，具体实现代码如下：

```
function updateCateFlag(cateid,flagVal){
    //alert("cateid:"+cateid + " flagVal:"+flagVal);
    var xmlhttp;
    try{
        xmlhttp = new XMLHttpRequest();
    }catch(e){
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    //创建请求结果处理程序
    xmlhttp.onreadystatechange = function(){
        if (4 == xmlhttp.readyState){
            if (200 == xmlhttp.status){
                var data = xmlhttp.responseText;
                //if(data) alert("修改成功");
            }else{
                alert("error");
            }
        }
    }

    //打开连接，true 表示异步提交
    xmlhttp.open("post", "/javascript.php?action=updateFlag", true);
    //当方法为 post 时需要按如下所示设置 http 头
    xmlhttp.setRequestHeader('Content-type','application/x-www-form-urle
                              ncoded');
    //发送数据
    xmlhttp.send("cateid=" + cateid + "&flagVal=" + flagVal);
}
```

该函数的核心代码为最后 3 句，前面几句是对 xmlhttp 的定义，原样复制即可。调用的 URL 是 javascript.php，其中在 open 时的?action=updateFlag 也可放在 send 中，如该句可改为：

```
xmlhttp.send("action=updateFlag&cateid=" + cateid + "&flagVal=" + flagVal);
```

我们再看看 javascript.php 对 action=updateFlag 的定义：

```
//修改 categary 的 flag 属性
```

```
if($action = 'updateFlag'){
    $cateid = intval($_POST['cateid']);
    $flagVal = intval($_POST['flagVal']);
    echo $cateid.$flagVal;

    updateFlag($cateid,$flagVal);
}

function updateFlag($cateid,$flagVal){
    global $db,$tablepre;
    $sql = "update ".$tablepre."categories set flag = ".$flagVal."
            where id=".$cateid;
    $db->query($sql);
}
```

当然，上面的代码也可简化，修改后的代码如下：

```
if($action = 'updateFlag'){
$cateid = intval($_POST['cateid']);
$flagVal = intval($_POST['flagVal']);
//echo $cateid.$flagVal;

$sql = "update ".$tablepre."categories set flag = ".$flagVal."
        where id=".$cateid;
$db->query($sql);
}
```

再返回去看看，Ajax 的实现也不是很复杂。但由于不需要操作页面上的内容，所以按这种方式来实现还是很简单；否则就会因为要解释返回的内容而变得复杂。可参考**“11.2 jQuery 入门”**后面的**“小知识：jQuery 为开发带来便利”**，那里讲述的项目最先用的就是这种方式，但在解释返回结果时很麻烦，而改用 jQuery 后，在很少时间内就实现了该功能。

4. 文章

“文章”菜单项是 HwCMS 的重点，用于对文章进行编辑，其实也只是对表 hw_articles (文章)的处理，如图 12-30 和图 12-31 所示。

id	istop	cateid	catename	catedir	title	htmlurl	abstract	content	tags
1	1	1	人生需要引导	plan	浩为成长计划		<MEMO>	<MEMO>	浩为资源堂 浩为成
2	0	1	人生需要引导	plan	人生需要引导-寄语		<MEMO>	<MEMO>	
3	1	1	人生需要引导	plan	浩为百校送书行动		<MEMO>	<MEMO>	浩为资源堂
4	0	1	人生需要引导	plan	编程快速入门-Java		<MEMO>	<MEMO>	
5	0	1	人生需要引导	plan	Java项目培训		<MEMO>	<MEMO>	
6	0	1	人生需要引导	plan	职场十分钟简介		<MEMO>	<MEMO>	
7	0	1	人生需要引导	plan	人生需要引导-引导		<MEMO>	<MEMO>	
8	0	1	人生需要引导	plan	从李开复离职创业		<MEMO>	<MEMO>	
9	0	1	人生需要引导	plan	浩为学习小组		<MEMO>	<MEMO>	
10	0	1	人生需要引导	plan	免费浩为QQ群简介		<MEMO>	<MEMO>	

图 12-30　“文章”列表

首页　设置　分类　文章　HTML　评论　标签　引用　模板　数据　备份　用户　注销

添加文章 管理文章　编辑

ID	标题	分类/目录	评论/引用	属性	操作
9	浩为学习小组	人生需要引导/plan	0/0	无	编辑 刷新
8	从李开复离职创业谈人生需要引导	人生需要引导/plan	0/0	无	编辑 刷新
7	人生需要引导-引导小知识	人生需要引导/plan	0/0	无	编辑 刷新
6	职场十分钟简介	人生需要引导/plan	0/0	无	编辑 刷新
5	Java项目培训	人生需要引导/plan	0/0	无	编辑 刷新
4	编程快速入门-Java	人生需要引导/plan	0/0	无	编辑 刷新
3	浩为百校送书行动	人生需要引导/plan	0/0	置顶	编辑 刷新
2	人生需要引导-寄语大学生	人生需要引导/plan	0/0	无	编辑 刷新
1	浩为成长计划	人生需要引导/plan	0/0	置顶	编辑 刷新

全选 管理选项：置顶　删除选中　取消置顶　移动

Records:114 8/8 « ‹ 4 5 6 7 8 › Last »

图 12-31　“文章”操作界面

“文章”菜单项的访问地址为 http://127.0.0.1/admin.php?m=articles&a=artlist，可见其功能实现文件为 D:\howwe\wwwroot\admin\articles.php，该文件先执行 `checklogin();`，检查是否已登录，再根据$a 的值进行处理。一共有 6 种情况：artlist、artadd、artsave、artedit、arteditsave 和 operator。具体代码请自己阅读。

HTML 文件为 D:\howwe\wwwroot\templates\admin\articles.php，代码格式与“首页”的一样，但会根据$a 的值显示不同的 HTML，共 4 种情况：artlist、artadd、artedit 和 move。在代码中已用空行隔开，以便阅读，具体情况请自己阅读。

图 12-31 中的“操作”有两种：

1）“刷新”能生成静态页面，范例 http://127.0.0.1/makehtml.php?a=refresh&artid=142。

2）“编辑”则可编辑该文章的内容，编辑窗口如图 12-32 所示。

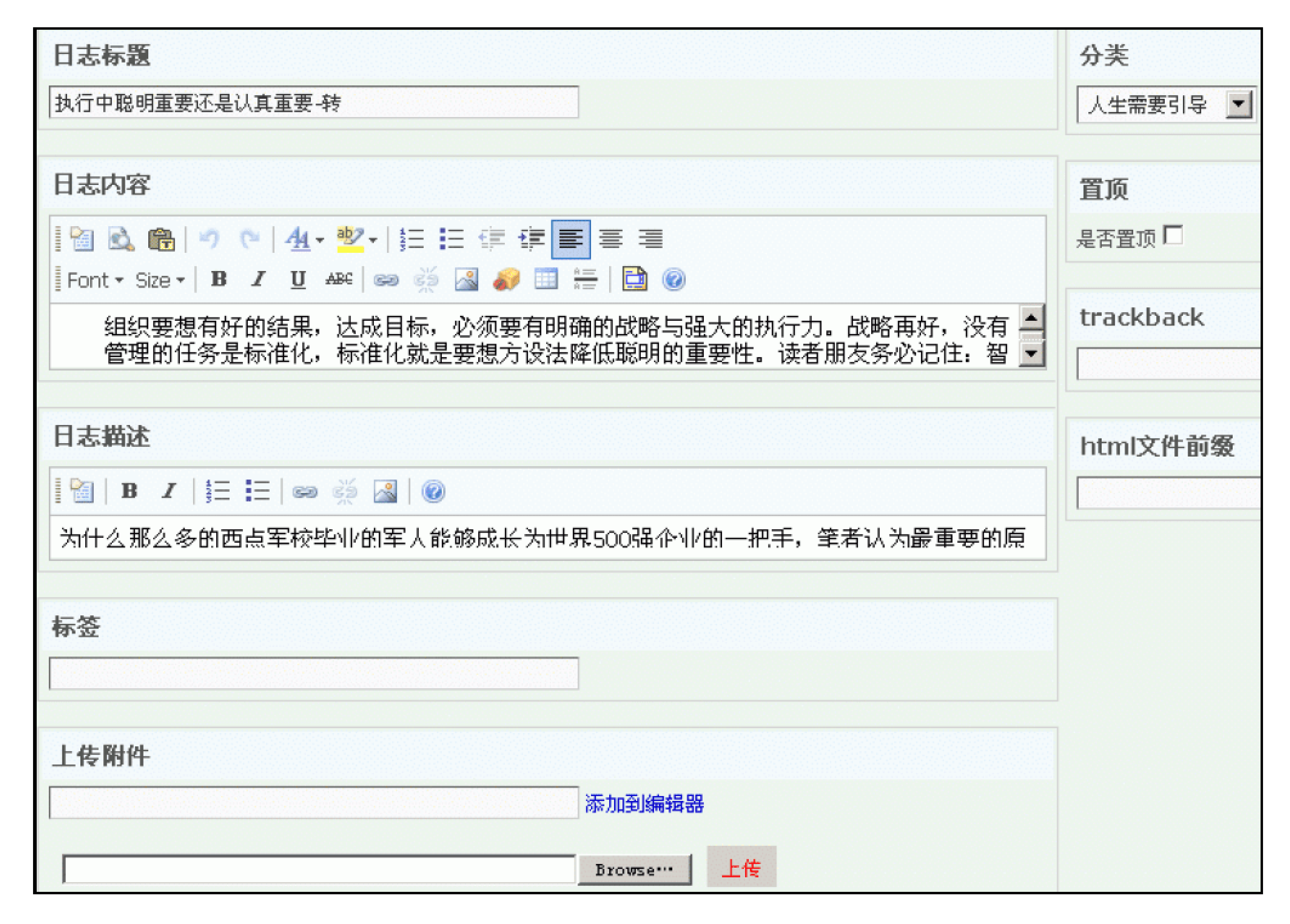


图 12-32　“文章”的编辑或添加界面

注意：

已对“日志内容”和“日志描述”两部分进行过调整高度操作。

5. HTML

“HTML”菜单项是 HwCMS 的重中之重，用于将文章生成为静态页面，如图 12-33 所示。

菜单项的访问地址为 http://127.0.0.1/admin.php?m=html&a=article，可见其功能实现文件为 D:\howwe\wwwroot\admin\html.php，该文件先执行 `checklogin();`，检查是否已登录，再读取分类信息和文章的最小及最大 id。具体代码请自己阅读。

HTML 文件为 D:\howwe\wwwroot\templates\admin\html.php，代码格式与“首页”的一样，但会根据`$a` 的值显示不同的 HTML，共两种情况：article 和 list。在代码中已用空行隔开以便阅读。具体情况请自己阅读。

图 12-33　“HTML”界面

“刷新”功能的实现文件为 makehtml.php，将文章生成为静态的 HTML 文件，刷新有三种方式：batchrefresh、accordingid 和 caterefresh，后者用来生成文章列表。

查看 makehtml.php 代码中 batchrefresh 与 accordingid 的实现，先从数据库中读出相应文章，再包含 show.php 文件，通过 writetofile 函数生成静态文件，放在相应文件夹下。而 caterefresh 对应的文件为 normal。

但这三种实现都没考虑目录不存在的情况，因为目录一般会存在，在建立分类的时候同时就建立了目录。不过，这可算是设计的一个不足。在设计时，应尽力多考虑可能存在的不足，并设法弥补，那样开发出来的系统在用户使用时出现问题的可能性才会比较低。

其他菜单项本书不再说明，请参照前面的方法自行阅读。

第 13 章

浩为资源堂代码修改详解

浩为资源堂基于 UCenter Home(以下简称为 UCH)，用户在使用它时很麻烦，有时会增加不少没有必要的时间浪费。为了方便用户的使用，我对原有结构进行了很大修改，使用户在发表日志时可指定群组，即在发表日志时，可同时发表在群组里，从而减少用户时间的浪费。

在软件开发时，尽可能地为用户的操作提供便利，这样才能更好地解决用户的问题，才能真正实现软件开发的目标。

注意：

运行本章提到的 URL 前，必须先运行 30-php-Start.bat，以加载 PHP 运行环境。

13.1　原 有 问 题

运行 http://127.0.0.1/q，出现的窗口中会有图 13-1 所示的登录栏。用户名、密码均为 admin。登录成功后，先提示“登录成功”，再出现浩为资源堂主页，部分截图如图 13-2 所示。



图 13-1　浩为资源堂-登录栏



图 13-2　浩为资源堂主页-已登录

如图 13-2 所示，浩为资源堂中“群组”下的内容很多，可见“群组”是重点，我们对比一下 UCH 的标准界面，如图 13-3 所示。



图 13-3　UCenter Home 标准界面

在 UCH 中，用户大多只会发表发表自己的日志，少数人还会在加入的群组中发发话题，因为发表的话题在群组中显示，所以整个群组的人都能看到，甚至还能“置顶”或设为“精华”。但是话题和日志没有任何关系，如果已经发表的日志想显示在群组中，就得重新发表，这将浪费用户的时间及热心。

图 13-4 所示为标准 UCH 日志，最新日志发表时间为 2010-07-10。图 13-5 为 UCH 群组，时间中没有 2010-07-10，可见日志和群组内容没有任何联系。

说明：图 13-3、图 13-4 和图 13-5 均截自 http://www.hunaner.net/home.php 湖南人在上海网站中的“家园”，该网站为湖南湘西的学生做了大量公益助学活动。详情请浏览“爱心公益”http://www.hunaner.net/forum-9102-1.html。希望有更多的人关心并帮助贫困学生。

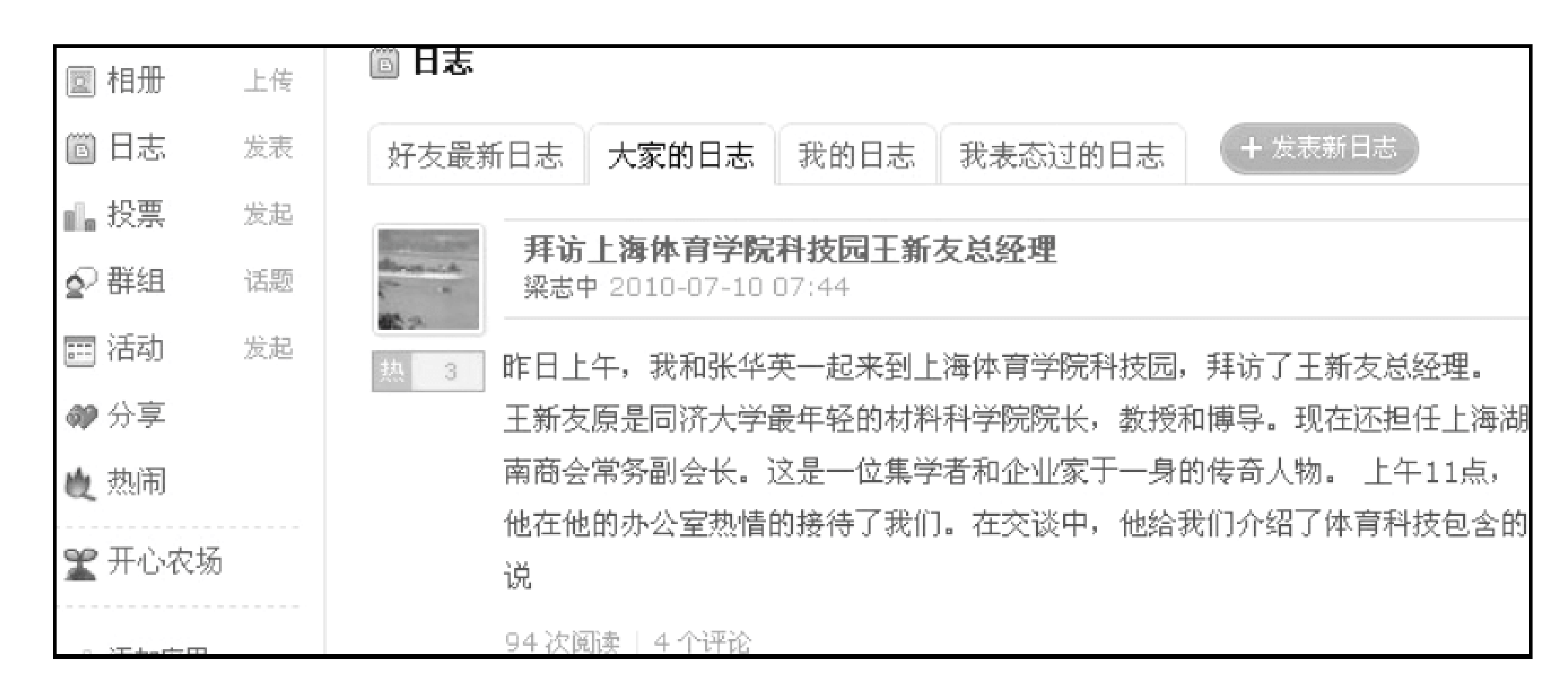


图 13-4　UCenter Home 日志



图 13-5　UCenter Home 群组

浩为资源堂现已改变这个状况，在发表日志时，可指定需要发表的群组；在群组里发表话题时可指定日志分类，不指定日志分类则直接放着日志下，以减少用户时间的浪费，从而为用户的使用提供方便。

图 13-6 显示的日志与图 13-4 相比，多了一个“分类”，如图 13-6 中的“人生五‘商’谈”一文属于“人生需要引导”分类(分类即群组)。在图 13-7 所示的群组中，可以看到前面提到的“人生五‘商’谈”。在编辑该文时，界面如图 13-8 所示，标题左边有个“日志分类”选择框，右边有个“群组分类”选择框，这样就可为用户带来方便。



图 13-6　浩为资源堂日志



图 13-7　浩为资源堂群组

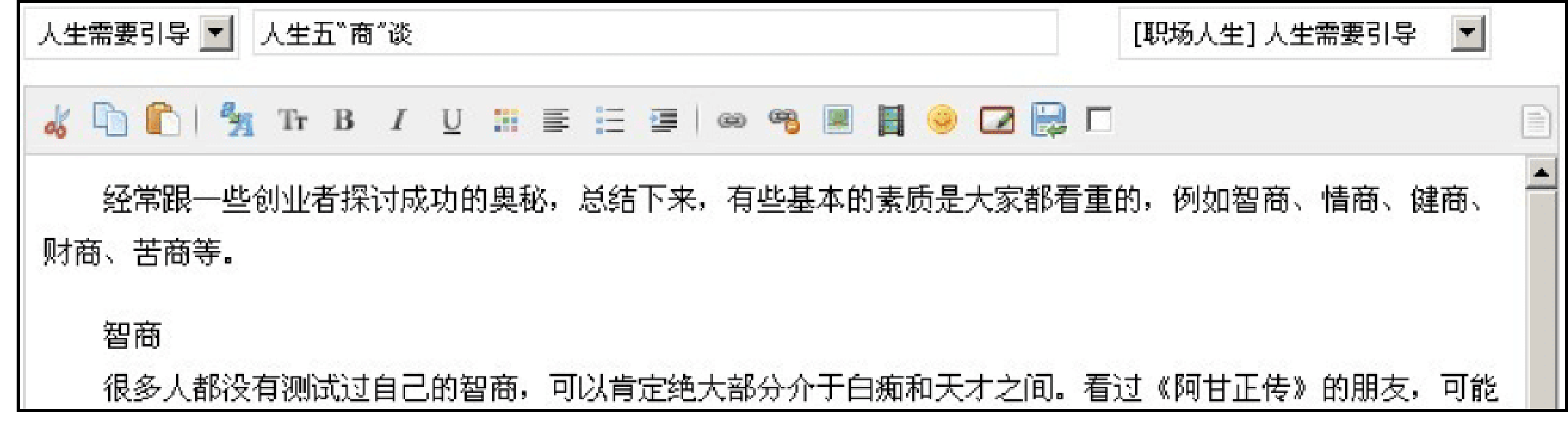


图 13-8　编辑文章

小知识：如何安装标准的 UCH

浩为资源堂采用的是日期为 2009-08-25 的 V2.0 版，文件为 UCenter_Home_2.0_SC_UTF8.zip，地址为 http://download.comsenz.com/UCenter_Home/2.0，但该地址已不再提供日期为 2009-08-25 的版本。为了能让读者对比这两种版本间的差别，本书特提供可供安装的原始版本。D:\howwe\wwwroot\home 下的 readme.txt 对 UCenter Home 有简单说明，下面详细示范安装步骤：

(1) 打开 UCcenter 用户管理中心的地址 http://127.0.0.1/home/uc，如图 13-9 所示，密码为 admin，输入验证码后，单击**登录**。



图 13-9　UCenter 用户管理中心

(2) 选择**应用管理**，再选择**添加新应用**，界面如图 13-10 所示，将 domainname 修改为 127.0.0.1/home 后单击**安装**，将出现如下提示信息：“您需要首先将程序根目录下面的

config.new.php 文件重命名为 config.php。”

(3) 将 D:\howwe\wwwroot\home 下的 config.new.php 文件重命名为 config.php，再单击浏览器的**刷新**按钮(也可按 F5 键)，按提示安装至图 13-11 所示界面。

提示：

安装包解压后的 upload 目录下的所有文件均在 home 目录中。



图 13-10　添加新应用

图 13-11　设置数据库信息

数据库用户名为 root，密码为 1，使用数据库为 test，输入后**确认**。

(4) 稍等片刻，出现图 13-12 所示界面，增加用户 admin，密码也为 admin。确认后出现图 13-13 所示界面，程序安装完成。

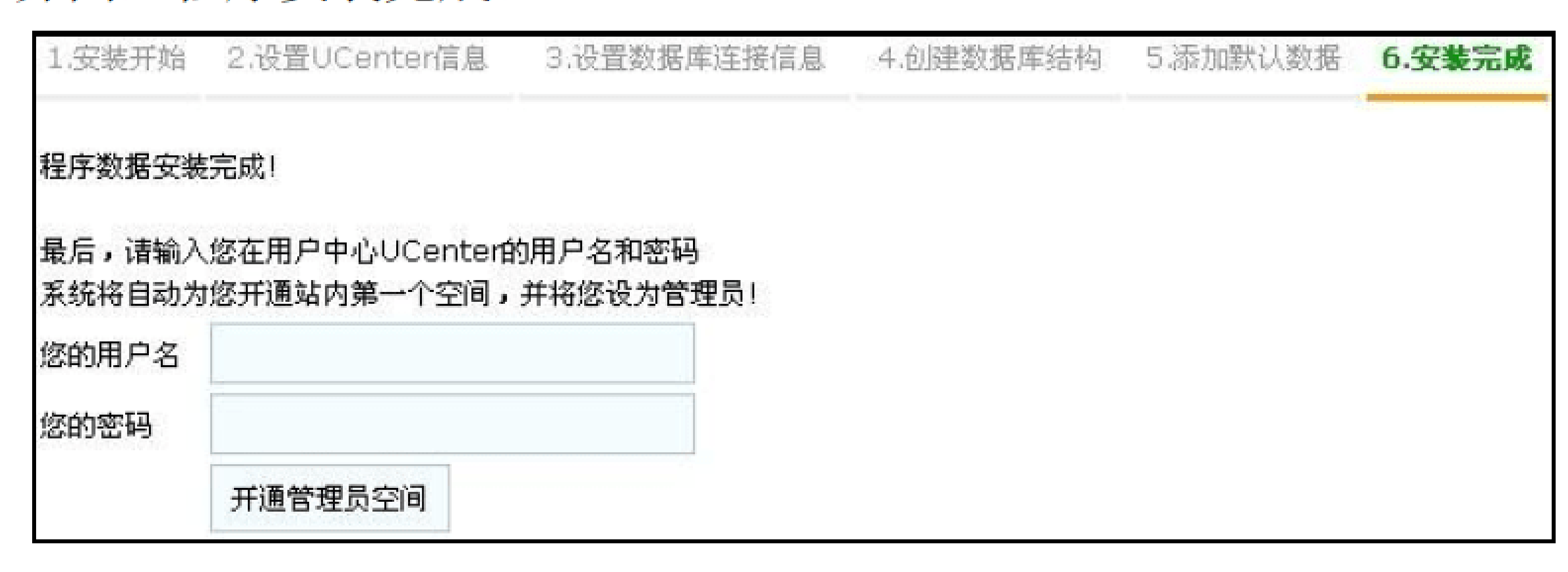


图 13-12　设置管理员及密码



图 13-13　安装完成后的提示

注意，如下两个链接地址请记住：

- 进入我的空间：网址为 http://127.0.0.1/home/space.php。进入我的主页，开始 UCenter Home 之旅。
- 进入管理平台：网址为 http://127.0.0.1/home/admincp.php。以管理员身份对站点参数进行设置。

在浏览器中打开 http://127.0.0.1/home，界面如图 13-14 所示。



图 13-14　UCenter Home 首页

13.2　修改数据库

通过仔细阅读 UCH 中“日志”和“群组”对应的页面，发现和“日志”相关的表有 q_blog(主表)、q_blogfield(从表)、q_comment(评论)、tagblog、class、friend 和 clickuser。前三个为主要表；而和“群组”相关的表有 q_thread(主表)、q_post(从表)、event、userevent、clickuser 和 mtag。群组的评论放在从表中，不像日志表那样单独放在评论表(q_comment)里。

尽管浩为资源堂中群组的内容较多，但用户使用的较多的是日志，所以对数据库的修改以日志表为主，将相应的“群组”表的字段添加在“日志”表里即可。我们先看看修改前的日志及群组表，参见图 13-15 至图 13-19。

注意：

浩为资源尝用的数据库为 a0903171509，标准 UCH 用的为 test，分别对应 D:\howwe\mysql\data 下的目录。

Name	Type	NULL	Default	Extras
Primary Index	blogid			unique
blogid	mediumint(8) unsigned	No	0	
uid	mediumint(8) unsigned	No	0	
tag	varchar(255)	No		
message	mediumtext	No		
postip	varchar(20)	No		
related	text	No		
relatedtime	int(10) unsigned	No	0	
target_ids	text	No		
hotuser	text	No		
magiccolor	tinyint(6)	No	0	
magicpaper	tinyint(6)	No	0	
magiccall	tinyint(1)	No	0	

图 13-15 日志表从表 q_blogfield

Name	Type	NULL	Default	Extras
Primary Index	blogid			unique
uid	uid,dateline			
topicid	topicid,dateline			
dateline	dateline			
blogid	mediumint(8) unsigned	No	<auto_increme...	
topicid	mediumint(8) unsigned	No	0	
uid	mediumint(8) unsigned	No	0	
username	char(15)	No		
subject	char(80)	No		
classid	smallint(6) unsigned	No	0	
viewnum	mediumint(8) unsigned	No	0	
replynum	mediumint(8) unsigned	No	0	
hot	mediumint(8) unsigned	No	0	
dateline	int(10) unsigned	No	0	
pic	char(120)	No		
picflag	tinyint(1)	No	0	
noreply	tinyint(1)	No	0	
friend	tinyint(1)	No	0	
password	char(10)	No		
click_1	smallint(6) unsigned	No	0	
click_2	smallint(6) unsigned	No	0	
click_3	smallint(6) unsigned	No	0	
click_4	smallint(6) unsigned	No	0	
click_5	smallint(6) unsigned	No	0	

图 13-16 日志表主表 q_blog

Name	Type	NULL	Default	Extras
Primary Index	cid			unique
authorid	authorid,idtype			
id	id,idtype,dateline			
cid	mediumint(8) unsigned	No	<auto_increme...	
uid	mediumint(8) unsigned	No	0	
id	mediumint(8) unsigned	No	0	
idtype	varchar(20)	No		
authorid	mediumint(8) unsigned	No	0	
author	varchar(15)	No		
ip	varchar(20)	No		
dateline	int(10) unsigned	No	0	
message	text	No		
magicflicker	tinyint(1)	No	0	

图 13-17 日志评论 q_comment

Name	Type	NULL	Default	Extras
Primary Index	tid			unique
tagid	tagid,displayorder,las…			
uid	uid,lastpost			
lastpost	lastpost			
topicid	topicid,dateline			
eventid	eventid,lastpost			
tid	mediumint(8) unsigned	No	<auto_increme…	
topicid	mediumint(8) unsigned	No	0	
tagid	mediumint(8) unsigned	No	0	
eventid	mediumint(8) unsigned	No	0	
subject	char(80)	No		
magiccolor	tinyint(6) unsigned	No	0	
magicegg	tinyint(6) unsigned	No	0	
uid	mediumint(8) unsigned	No	0	
username	char(15)	No		
dateline	int(10) unsigned	No	0	
viewnum	mediumint(8) unsigned	No	0	
replynum	mediumint(8) unsigned	No	0	
lastpost	int(10) unsigned	No	0	
lastauthor	char(15)	No		
lastauthorid	mediumint(8) unsigned	No	0	
displayorder	tinyint(1) unsigned	No	0	
digest	tinyint(1)	No	0	
hot	mediumint(8) unsigned	No	0	
click_11	smallint(6) unsigned	No	0	
click_12	smallint(6) unsigned	No	0	
click_13	smallint(6) unsigned	No	0	
click_14	smallint(6) unsigned	No	0	
click_15	smallint(6) unsigned	No	0	

图 13-18　群组表主表 q_thread

Name	Type	NULL	Default	Extras
Primary Index	pid			unique
tid	tid,dateline			
pid	int(10) unsigned	No	<auto_increme…	
tagid	mediumint(8) unsigned	No	0	
tid	mediumint(8) unsigned	No	0	
uid	mediumint(8) unsigned	No	0	
username	varchar(15)	No		
ip	varchar(20)	No		
dateline	int(10) unsigned	No	0	
message	text	No		
pic	varchar(255)	No		
isthread	tinyint(1)	No	0	
hotuser	text	No		

图 13-19　群组表从表 q_post

对比图 13-16 和图 13-18，可知后者比前者多一些字段，两者字段分别如下：

q_blog (5 个字段一行)：

```
blogid, topicid, uid, username, subject,
classid, viewnum, replynum, hot, dateline,
```

```
pic, picflag, noreply, friend, password,
click_1, click_2, click_3, click_4, click_5
```

q_thread：

```
tid, topicid, tagid, eventid, subject,
magiccolor, magicegg, uid, username, dateline,
viewnum, replynum, lastpost, lastauthor, lastauthorid,
displayorder, digest, hot, click_11, click_12,
click_13, click_14, click_15
```

其中：表 q_blog 的主关键字为 blogid，表 q_thread 的为 tid，故 tid 对应 blogid；多出的字段如图 13-20 所示，多出的索引如图 13-21 所示，其中 uid1 对应于群组表中 uid：

tagid	mediumint(8) unsigned	No	0
eventid	mediumint(8) unsigned	No	0
magiccolor	tinyint(6) unsigned	No	0
magicegg	tinyint(6) unsigned	No	0
lastpost	int(10) unsigned	No	0
lastauthor	char(15)	No	
lastauthorid	mediumint(8) unsigned	No	0
displayorder	tinyint(1) unsigned	No	0
digest	tinyint(1)	No	0

图 13-20　群组表主表比日志主表多出的字段

tagid	tagid,displayorder,lastpost
uidl	uid,lastpost
lastpost	lastpost
eventid	eventid,lastpost

图 13-21　群组表主表比日志主表多出的索引

对比从表，分别是图 13-15 和图 13-19，仅两个字段不同，如图 13-22 所示，最后两个字段为新增字段。

Name	Type	NULL	Default	Extras
Primary Index	blogid			unique
blogid	mediumint(8) unsigned	No	0	
uid	mediumint(8) unsigned	No	0	
tag	varchar(255)	No		
message	mediumtext	No		
postip	varchar(20)	No		
related	text	No		
relatedtime	int(10) unsigned	No	0	
target_ids	text	No		
hotuser	text	No		
magiccolor	tinyint(6)	No	0	
magicpaper	tinyint(6)	No	0	
magiccall	tinyint(1)	No	0	
pic	varchar(255)	No		
isthread	tinyint(1)	No	0	

图 13-22　群组表从表比日志主表多出的字段

对数据表进行修改之后，接下来是代码的修改。为了更好地了解修改的过程，我们先讲述相关的知识。

小知识：UC Home 数据库部分表说明

UC Home 2.0 共 78 个表，其中 q_adminsession 与 q_session 为内存表，其余表为 MyISAM。以下是部分表的简单介绍：

1) q_member　站点成员表，存放站点成员的临时密码信息
2) q_ad　广告表，存放广告设置信息
3) q_adminsession　用户管理 session 表，存放用户登录后台管理的 session
4) q_album　相册表，存放用户相册信息
5) q_app　应用列表
6) q_block　数据调用模块表，存放数据调用的信息
7) q_blog　用户日志表，存放用户日志标题等信息
8) q_blogfield　日志附加字段表，存放用户日志内容等信息
9) q_cache　缓存表，存放缓存信息
10) q_class　分类表，存放分类信息
11) q_comment　评论回复留言信息表，存放用户留言、评论、回复
12) q_docomment　记录表，记录及记录回复
13) q_config　站点配置信息表，存放站点配置信息
14) q_cron　更新浏览数统计、清理过期的 feed、清理个人通知
15) q_data　数据信息表，存放站点临时数据信息
16) q_doing　记录表，存放用户记录信息
17) q_feed　feed 表，存放 feed 信息
18) q_friend　好友表，存放好友信息
19) q_invite　邀请表，存放好友邀请信息
20) q_log　用户查看数缓存表，存放用户空间查看数、日志查看数的缓存信息
21) q_mailcron　邮件任务表，存放将要发邮件的 email 或用户 ID
22) q_mailqueue　邮件队列表，存放将要发送的邮件标题、内容等信息
23) q_mtaginvite　存放群组邀请信息
24) q_mtag　群组表，存放群组信息
25) q_myapp　存放漫游应用信息
26) q_myinvite　存放漫游邀请
27) q_notification　通知表，存放用户通知
28) q_pic　图片表，存放用户图片
29) q_poke　招呼表，存放用户打招呼信息
30) q_post　话题和话题回复表，存放话题的内容和话题回复

31) q_profield　　群组栏目设置表，存放群组栏目的设置信息
32) q_profilefield　用户栏目表，存放用户栏目设置信息
33) q_report　　用户举报表
34) q_session　　用户 session 表，存放用户 session 信息
35) q_share　　分享表，存放用户分享信息
36) q_show　　存放排行榜信息
37) q_space　　个人空间表，存放用户个人空间积分好友数等信息
38) q_spacefield　个人空间附加表，存放用户个人空间主题、个人策略等附加信息
39) qq_spacelog　用户空间变化信息表
40) q_tag　　关键字表
41) q_tagblog　　日志和 tag 对应关系表，存放 blog 对应 tag 关系信息
42) q_tagspace　用户群组对应关系表，存放应用用户和群组的对应关系
43) q_thread　　群组话题表，存放群组话题标题等信息。话题主表
44) q_trace　　用户脚印信息表，存放用户脚印记录
45) q_userapp　应用表，存放应用信息
46) q_usergroup　用户组表，存放用户组权限设置信息会员分类
47) q_userlog　　用户 log 表，存放用户更新等信息
48) q_usertask　用户任务表
49) q_visitor　　访客表，存放访客信息

13.3 模板技术

在第 12 章，我们已对模板技术做了简单讲述。模板技术通过模板引擎来实现，可模板引擎到底是什么呢？“引擎”两字一般都会让人觉得很高深，如游戏 3D 引擎、PHP 的 Zend 引擎确实很高深，但“模板引擎”这 4 个字只是用来唬人的、骗外行人的，以至于很多人在初学 PHP 时不敢去学这方面的技术。

在 PHP 里，模板引擎扮演着 View 的角色，通俗一点说只是页面，但页面是一个很重要的角色，是最终用户能看到的系统界面(UI)，用户用此界面来实现与系统交互，从而完成用户的工作。正如第 12 章所说的那样，模板技术只是为了使页面不出现 PHP 代码，而另外创造的用来代替 PHP 代码的编码，然后通过模板引擎解释出对应的 PHP 代码，这是模板技术的本质。再简单一点来说，模板技术拥有一套自己的编码规则，用于替换模板中本该是 PHP 代码的代码，在运行时再由模板引擎解释成 PHP 代码。

Smarty(http://www.smarty.net/)是 PHP 的官方模板引擎，是一个基于 PHP 开发的 PHP 模板引擎，使 PHP 代码与外在内容(HTML 页面)分离，换种说法是，逻辑与外在内容的分离，但这种说法是有问题的，因为 HTML 页面中还是包含了很多通过模板语句而实现的逻

辑，其**实质只是 HTML 页面中的 PHP 代码用模板语句代替了而已**。

模板的目的只是让使用 PHP 的程序员同美工分离：程序员改变程序的逻辑内容而不会影响美工的页面设计，美工重新修改页面不会影响到程序的程序逻辑。这在多人合作的项目中特别重要。但模板的使用会降低运行效率，正如在 1.1 节中介绍的那样，因为 PHP 是一种“**每次请求就是一个完整的生命周期**”的语言，每请求一次，所有相关对象都必须重新加载，采用模板技术也就是每次都得加载模板引擎。举个例子，本来某一个功能的实现只要 100 句代码，即加载时只需要加载 100 句，但一加入模板引擎，需要加载的语句可能上千句，速度能不受影响吗？

在 Smarty 网站上，有详尽的用户手册，可选择在线 HTML 或 PDF 版本，具体我们不再展开讨论。只简单介绍一下其原理：接到请求后，先判断是否是第一次请求该 URL，如果是，将该 URL 所需的模板文件“编译”成 PHP 代码，然后返回；如果不是，说明该 URL 的模板已经被“编译”过了，检查不需要重新编译后，马上返回。重新编译的条件可以自己设定，可设为固定时限，但默认是模板文件被修改。

Smarty 常见问题如下：

1) 提示说找不到所需文件

并不是每一个人都会按照 Smarty 默认的目录结构来写应用，故需要手工指定目录，假设目录结构如图 13-23 所示：

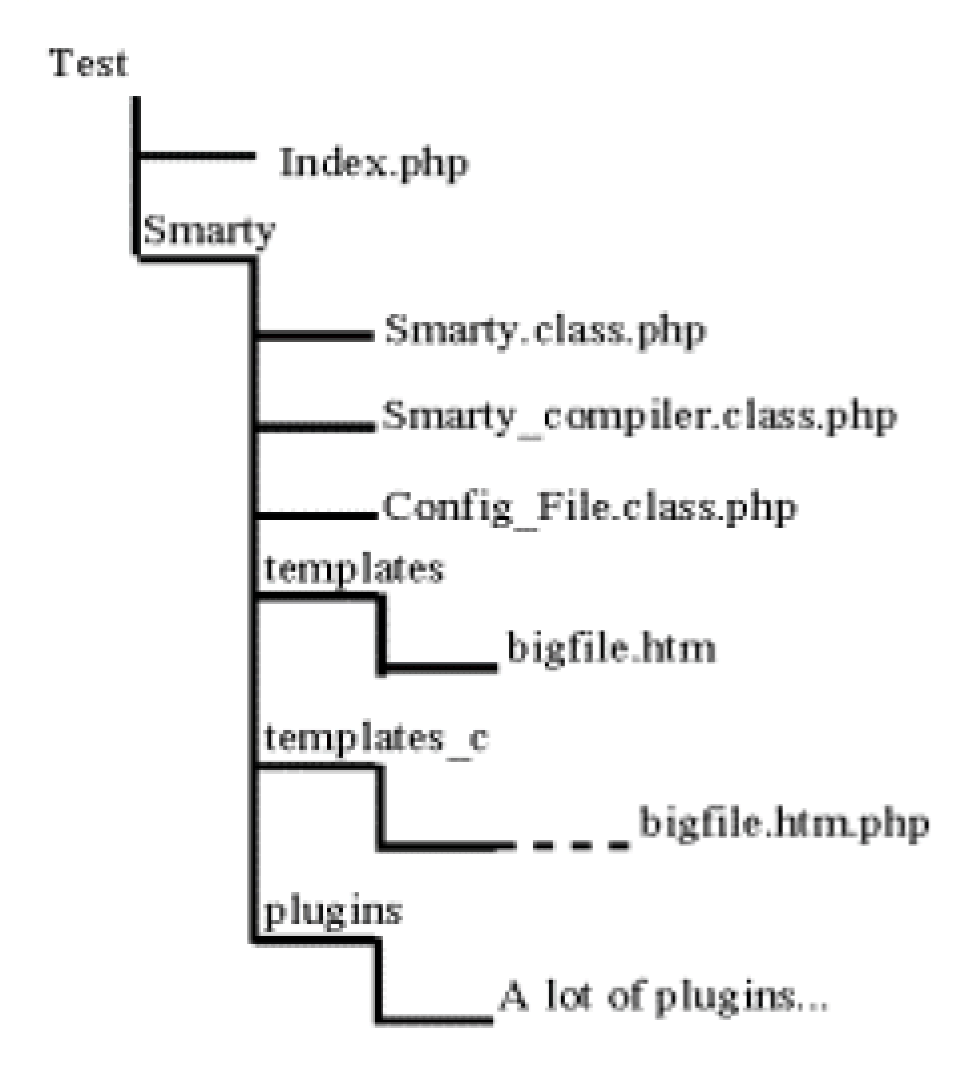


图 13-23　Smarty 范例目录结构

则需要在 index.php 里指定目录结构：

```
$smart->template_dir = "smarty/templates/";
$smart->compile_dir = "smarty/templates_c/";
$smart->config_dir = "smarty/configs/";
$smart->cache_dir = "smarty/cache/";
```

2) 有时用 Dreamweaver 生成的模板不能用

并不是模板文件问题，而是因为 Smarty 默认的标记分隔符是{}，而 JavaScript 也包含这个标记。所以只得指定为其他分隔符，代码如下：

```
$smart->left delimiter = "{/";
$smart->right_delimiter = "/}";
```

在 Smarty-3.0 RC 3 中不需要设置分隔符。以下是官方下载包的范例：

```
<?php
require('../libs/Smarty.class.php');

$smarty = new Smarty;

//$smarty->force_compile = true;
$smarty->debugging = true;
$smarty->caching = true;
$smarty->cache_lifetime = 120;

$smarty->assign("Name","Fred Irving Johnathan Bradley Peppergill",true);
$smarty->assign("FirstName",array("John","Mary","James","Henry"));
$smarty->assign("LastName",array("Doe","Smith","Johnson","Case"));
$smarty->assign("Class",array(array("A","B","C","D"),array("E", "F", "G",
        "H"), array("I", "J", "K", "L"), array("M", "N", "O", "P")));

$smarty->assign("contacts", array(array("phone" => "1", "fax" => "2", "cell"
         => "3"), array("phone" => "555-4444", "fax" => "555-3333", "cell"
         => "760-1234")));

$smarty->assign("option_values", array("NY","NE","KS","IA","OK","TX"));
$smarty->assign("option_output", array("New York","Nebraska","Kansas",
         "Iowa","Oklahoma","Texas"));
$smarty->assign("option_selected", "NE");

$smarty->display('index.tpl');
?>
```

总之，Smarty 不难，自己开发时“照葫芦画瓢”即可。

小知识：10 个 PHP 模板引擎

1) Smarty ——http://www.smarty.net/

Smarty 的特点是将模板编译成 PHP 脚本，然后执行这些脚本。

2) Rtemplate ——http://www.phpguru.org/downloads/rtemplate/

一个非常容易使用、功能强大并且快速的模板引擎，它帮助你把页面布局和设计从代码中分离。

3) ShellPage ——http://www.maiatech.com/varpage.php

一个简单易用的类，可以让你的整个网站布局基于模板文件，修改模板就能改变整个站点。

4) STP Simple Template Parser ——http://www.script.gr/sc/scripts/STP/

一个简单、轻量级并且易于使用的模板分析类。它可以从多个模板中组装一个页面，把结果页面输出到浏览器或文件系统。

5) OO Template Class ——http://www.net-track.ch/opensource/template/

一个可以用在自己程序中的面向对象的模板类。

6) Savant ——http://phpsavant.com/

一个强大且轻量级的 PEAR 兼容模板系统。它是非编译型的，使用 PHP 语言本身作为它的模板语言。

7) AvanTemplate ——http://avantemplate.sourceforge.net/

多字节安全的模板引擎，占用很少的系统资源。它支持变量替换，内容块可以设置为显示或隐藏。

8) Grafx's Fast Template ——http://www.grafxsoftware.com/product.php?id=26

一个修改版本的 Fast Template 系统，它包括缓存功能以及调试控制台等功能。

9) TemplatePower ——http://templatepower.codocad.com/

一个快速、简单、功能强大的模板类。主要功能有嵌套的动态块支持，块/文件包含支持以及显示/隐藏未赋值的变量。

10) TinyButStrong ——http://www.tinybutstrong.com/

一个支持 MySQL、ODBC、SQL-Server 和 ADODB 的模板引擎。它包含 7 个方法和两个属性。

13.4 修改详细介绍

UCH 2.0 的代码从总体来说就一个字：乱，简直就是很多代码的堆积！代码的实现没有统一的标准，开发人员想怎么写就怎么写，就写成了各自的风格，最后再堆积起来。这是我在修改时的第一感觉，正因为其代码的混乱，有些修改实在没法按原来的构思去改，最后只得变通变通，如对原有群组话题的修改，改了几次还是有问题，最后只得以日志的形式来改。

我很认同一位朋友的话：如果真想学习 PHP 编程，建议多看看 phpbb(phpbb 官方网站为 http://www.phpbb.com/)。看其实现，层次清晰，像是艺术品；不像国内的很多代码，简直是一堆草。是草，还是艺术品？给人带来的感觉完全不同：艺术品越看越喜欢，是一种

欣赏；而草呢，杂草从生，有时无从下手，自然越看越没心情。

下面先介绍一下 UCH 的基本情况：

1. 模板技术

打开 D:\howwe\wwwroot\q\index.php，第 62 行为引入页面的代码：`include_once template("index");`，如图 13-24 所示(这里采用 UltraEdit 来查看)。

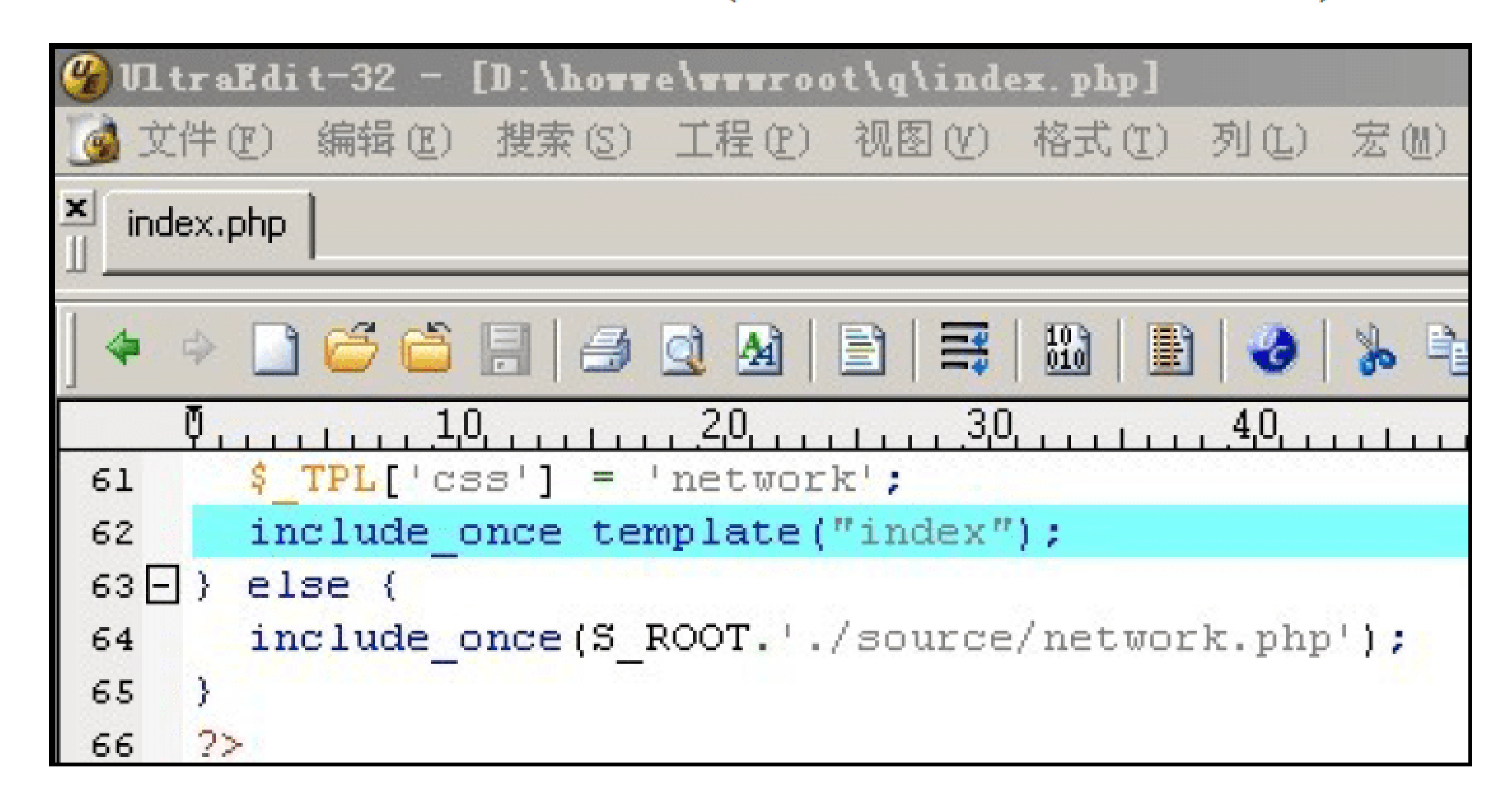


图 13-24　浩为资源堂引用模板

搜索“function template(”，在文件 D:\howwe\wwwroot\q\source\function_common.php 中有其定义，函数代码如下(已在相应代码后添加注释)：

```
//模板调用
function template($name) {
    global $_SCONFIG, $_SGLOBAL;

    if($_SGLOBAL['mobile']) {    //判断是否为手机版，如果是，则直接用模板文件
        $objfile = S_ROOT.'./api/mobile/tpl_'.$name.'.php';
        if (!file_exists($objfile)) {
            showmessage('m_function_is_disable_on_wap');
        }
    } else {
        if(strexists($name,'/')) {   //如果包含“/”，则为指定目录下的文件，否
                                         则取模板文件夹下的文件
            $tpl = $name;
        } else {
            $tpl = "template/$_SCONFIG[template]/$name";
        }
        $objfile = S_ROOT.'./data/tpl_cache/'.str_replace('/','_',
                       $tpl).'.php'; //模板缓存文件地址，预编译后的模板文件存放处
        if(!file_exists($objfile)) { //不存在预编译文件，重新编译
```

```
                include_once(S_ROOT.'./source/function_template.php');
                parse_template($tpl);
            }
        }
        return $objfile;//返回模板缓存文件的地址
    }
```

注意：

为了能使用模板技术，必须包含文件：include_once('./common.php');。common.php 中加载了 ver.php、config.php、source/function_common.php、source/class_mysql.php、data/data_config.php、data/data_app.php、data/data_userapp.php、data/data_ad.php、data/data_magic.php 等，需要加载的代码行数预计为 2 500 行。可见其运行效率不会太高。

2. 目录结构

admin：管理后台，功能实现文件及模板文件放在该目录下。

api：扩展功能。

attachment：存放附件。

data：数据库以及和数据相关的配置参数，其中 tpl_cache 下面存放模板文件编译后的文件。

image：图片及相关文件。

language：语音包。

source：功能实现文件。

template：模板文件。

theme：风格。

uc_client：连接用户中心的客户端，浩为资源堂的用户中心在 D:\howwe\wwwroot\img\api 下，通过 http://127.0.0.1/img/api 来访问。

admincp.php：管理后台的控制文件，MVC 中的 C，以下简称为 C。

cp.php：编辑控制文件 C。

do.php：部分功能实现控制文件 C。

editor.php：文本编辑器代码。

index.php：主页。

magic.php：道具等效果的控制文件 C。

space.php：前台控制文件 C。

为了简化日志、群组、群组话题的 URL，分别将 space.php 的相关功能单独做成文件，分别是 me.php、ok.php、q.php，并可以直接使用类似于“?1”，即问号后直接跟编号的访问方式，例如 http://127.0.0.1/q/me.php?62 为“资源堂必读”。

3. 功能实现简介

在介绍目录结构时，介绍了几个控制文件，而控制文件就是功能实现的分配器，即通过此文件将相应的功能包含进来，完成相关的功能。

打开 space.php，在第 25 行有简单说明。

```
$dos = array('feed', 'doing', 'mood', 'blog', 'album', 'thread', 'mtag',
        'friend', 'wall', 'tag', 'notice', 'share', 'topic', 'home', 'pm',
        'event', 'poll', 'top', 'info', 'videophoto');
```

功能有 20 项，对应的功能实现文件在 D:\howwe\wwwroot\q\source 下，命名规则为 space_***.php，***为具体功能的名称，例如 space_feed.php。可打开 me.php、ok.php、q.php，并与 space.php 对比。

打开 cp.php，在第 12 行有简单说明。

```
$acs = array('space', 'doing', 'upload', 'comment', 'blog', 'album',
        'relatekw','common','class','swfupload', 'thread', 'mtag', 'poke',
        'friend','avatar', 'profile', 'theme', 'import', 'feed', 'privacy',
        'pm', 'share', 'advance', 'invite','sendmail','userapp', 'task',
        'credit', 'password', 'domain', 'event', 'poll',
        'topic','click','magic', 'top', 'videophoto');
```

功能有 37 项，对应的功能实现文件在 D:\howwe\wwwroot\q\source 下，但只有 36 个功能文件，命名规则为 cp_***.php，***为具体功能的名称，例如 cp_space.php。

打开 D:\howwe\wwwroot\q\source 下的 cp_space.php，代码实现比较简单，这里不再展开。请注意 `include_once template("cp_space");`为加载模板文件 cp_space，模板在 D:\howwe\wwwroot\q\template\default 下，对应文件为 cp_space.htm，代码如下：

```
<!--{template header}-->

<!--{if $_GET['op'] == 'delete'}-->

<h1>隐藏应用</h1>
<a href="javascript:hideMenu();" class="float_del" title="关闭">关闭</a>
<div class="popupmenu_inner" id="  userappform_{$_GET[appid]}">
    <form method="post" id="userappform_{$_GET[appid]}"
       name="userappform_{$_GET[appid]}" action="cp.php?ac=space&op=">
        <p>确定要隐藏该应用吗？</p>
        <p class="btn_line">
```

```
                <input type="hidden" name="refer" value="$_SGLOBAL[refer]">
                <input type="hidden" name="appid" value="$_GET[appid]" />
                <input type="hidden" name="type" value="$_GET[type]" />
                <!--{if $_SGLOBAL[inajax]}-->
                <input type="hidden" name="delappsubmit" value="true" />
                <button name="delappsubmit_btn" type="button" class="submit"
                  value="true" onclick="ajaxpost('userappform_
                  {$_GET[appid]}', 'userapp_delete', 2000)">确定</button>
                <!--{else}-->
                <button name="delappsubmit" type="submit" class="submit"
                  value="true">确定</button>
                <!--{/if}-->
            </p>

            <input type="hidden" name="formhash" value="<!--{eval echo
              formhash();}-->" />
        </form>
    </div>
    <!--{/if}-->

    <!--{template footer}-->
```

代码说明：

1) `<!--{template header}-->`用来加载头文件 header.htm，其中`<!--{`对应于 PHP 中的`<?php`，`}-->`对应于`?>`。

2) `<!--{if $_GET['op'] == 'delete'}-->`等同于 `if $_GET['op'] == 'delete'`，用来进行逻辑判断。

3) `<!--{template footer}-->`用来加载页脚文件 footer.htm。

总的来说，模板的编码规则不复杂，仅仅是相应 PHP 代码的另一种表现形式而已，不用看得很高深。正如前面所说，模板技术只是来吓唬外行人的，明白其原理后，就会觉得很简单，这也是我大力提倡“IT 简单化”的重要原因：**还原 IT 的本来面貌，不再让人觉得 IT 高不可攀**。

从本质上来说，UCH 的模板技术除了有对模板的预编译之外，和 HwCMS 的模板没啥区别，但 HwCMS 方式的效率会大大提高，因为 HwCMS 加载的代码很少。

对于 do.php 和 magic.php，请自己按同样的方式来分析，本书不再展开。

4. 头文件 header.htm 的修改

UCH 的大部分页面均由头文件 header.htm、页脚文件 footer.htm 和具体页面功能构成。范例如图 13-25 所示，顶部的横条和左侧的菜单条及公告(广告)部分由头文件产生。下面介绍如何将标准界面修改为图 13-26 所示的浩为资源堂主页。



图 13-25　UCenter Home 标准界面



图 13-26　浩为资源堂主页-已登录

使用 UltraEdit 打开 D:\howwe\wwwroot\q\template\default 下的 header.htm，如图 13-27 所示。UCH 原版可从 http://u.discuz.net 下载。header.htm 文件位置在目录 upload\template\default 下。

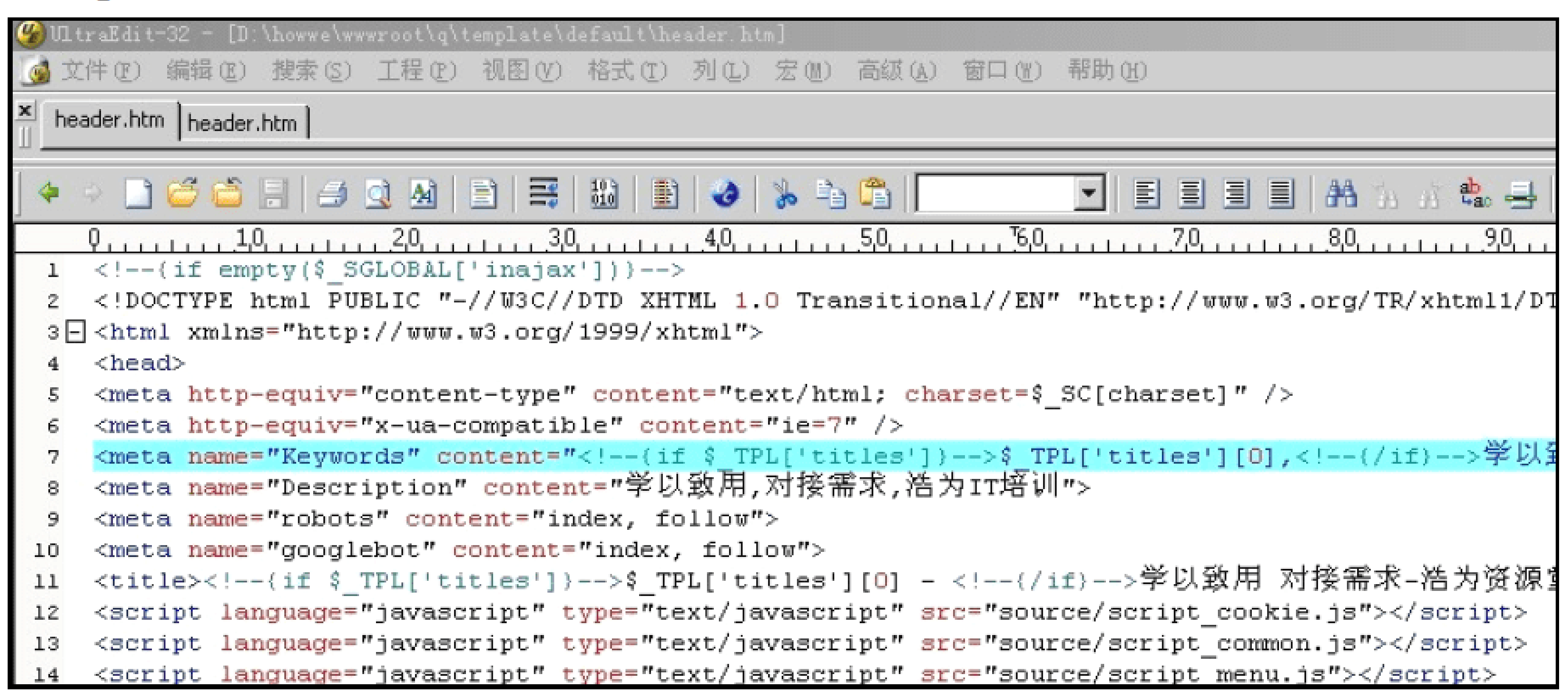


图 13-27　使用 UltraEdit 打开 header.htm

以下内容为对已修改内容的说明：

1) 添加两个 meta 用于 SEO 优化，位置在第 7、8 两行。

```
    <meta name="Keywords" content="<!--{if
$_TPL['titles']}-->$_TPL['titles'][0],<!--{/if}-->学以致用,对接需求,浩为 IT 培训,
浩为资源堂">
    <meta name="Description" content="学以致用,对接需求,浩为 IT 培训">
```

2) 修改 title 标题，将标题简化，位置在第 11 行。

```
    <title><!--{if $_TPL['titles']}-->$_TPL['titles'][0] - <!--{/if}-->学以致
用 对接需求-浩为资源堂 浩为 IT 培训，资源整合</title>
```

3) 添加顶部的广告，位置在第 40 行。

```
    <div id="ad_header"><a href="http://howwe.net" target="_blank"><img
src="image/howwetop.jpg" width="966" border="0" alt="引导学思想开创 IT 培训新模式,
以多层次 Java 培训为起点，知道自己要做什么"></a></div>
```

4) 删除“站内导航”代码，原版第 49~57 行，即<!--{if $_SGLOBAL['appmenu']}-->部分。

5) 修改顶部横条内容：将原 LOGO 修改为“浩为资源堂”图标；修改“首页”提示；添加菜单“群组”和“日志”，链接分别为“ok.php”、“me.php?view=all&orderby=dateline”。位置在第 42~46 行。

```
        <h1 class="logo"><!--{if $_SGLOBAL[supe_uid]}--><a
href="http://zyt.howwe.net/" title="引导学思想开创 IT 培训新模式，以多层次 Java 培训
为起点，知道自己要做什么。"><!--{else}--><a href="index.php" title="浩为资源堂
"><!--{/if}--><img src="template/$_SCONFIG[template]/image/logo.gif"
alt="$_SCONFIG[sitename]" /></a></h1>
            <ul class="menu">
                <li><!--{if $_SGLOBAL[supe_uid]}--><a href="home.php" title="
浩为资源堂，你的生活因此而精彩；学以致用，以 IT 技术、职场体验为主，面向大学生的个性化社区式
资源圈。"><!--{else}--><a href="index.php" title="浩为资源堂"><!--{/if}-->首页
</a></li>
                <li><a href="ok.php">群组</a></li>
                <li><a href="me.php?view=all&orderby=dateline">日志</a></li>
```

6) 修改左侧菜单条：将“日志”提到最前面，链接为 me.php；“群组”排第二，再添加 11 个子项，链接为 q.php；将“相册”排第三。并设置“limt”、“lim”、“limb”三个 CSS 样式以调整菜单间距，位置在第 93~105 行。

```
                <li><img src="image/app/blog.gif"><a
href="me.php?view=all&orderby=dateline">日志</a><em><a href="cp.php?ac=blog"
```

```
class="gray">发表</a></em></li>
                    <li id="limt"><img src="image/app/mtag.gif"><a
href="ok.php">群组</a><em><a href="cp.php?ac=thread" class="gray">话题
</a></em></li>
                    <li id="lim"> <a href="q.php?1">Java</a></li>
                    <li id="lim"> <a href="q.php?2">PHP</a></li>
                    <li id="lim"> <a href="q.php?4">数据库应用</a></li>
                    <li id="limb"> <a href="q.php?3">C/C++</a> <a
href="q.php?5">Net</a></li>
                    <li id="lim"> <a href="q.php?18">谈谈灌灌</a></li>
                    <li id="lim"> <a href="q.php?17">活动管理区</a></li>
                    <li id="limb"> <a href="q.php?19">浩为助学</a></li>
                    <li id="lim"> <a href="q.php?22" title="思维导图加强思考的
条理性，使你更快、更准的抓住问题点，从而迅速解决问题。">思维导图</a></li>
                    <li id="lim"> <a href="q.php?21">考研专栏</a></li>
                    <li id="limb"> <a href="q.php?7">人生需要引导</a></li>
                    <li><img src="image/app/album.gif"><a
href="space.php?do=album">相册</a><em><a href="cp.php?ac=upload"
class="gray">上传</a></em></li>
```

7) 修改公告(广告)部分(id="ad_contenttop")为浩为资源堂的推荐文章列表，位置在第 162 行。注意，已使用“floatleft”和“floatright”两个 CSS 样式将文章标题分开。

```
    <div id="ad_contenttop"><div class="floatleft"><a href="me.php?62" title="
浩为资源堂使用说明">资源堂必读</a> <a href="me.php?203">快速入门系列&电子档</a> <a
href="me.php?9">IT 入门指南</a> <a href="me.php?63">寄语大学生</a> <a
href="me.php?64" title="人的成长就是一个资源整合的过程">资源整合</a> <a
href="me.php?65" title="人生需要引导，简称为“引导学”，在于找回人生的本质，让人知道自
己要做什么。">引导学概述</a> <a href="me.php?66" title="汇集浩为资源堂精华，重发展示
自己的个性。浩为资源堂，你的生活因此而精彩">浩为资源堂月刊</a></div><div
class="floatright"><a href="http://howwe.net/bk1" title="《从人生需要引导论 Java
快速开发》相关文章列表，“人生需要引导”系列首册">《Java 快速开发..》</a> <a
href="http://howwe.net/bk2" title="《喻世明言新篇-人生需要引导》相关文章列表，“人生
需要引导”系列第二册">《喻世明言新篇..》</a></div></div>
```

图 13-28 为原始版与修改版的对比，使用步骤：在 UltraEdit 分别打开原始版和修改版 header.htm，再单击 UltraEdit 工具栏中的“比较文件”图标(图标上为 UC 两个字母的组合，图中的箭头所指之处)。

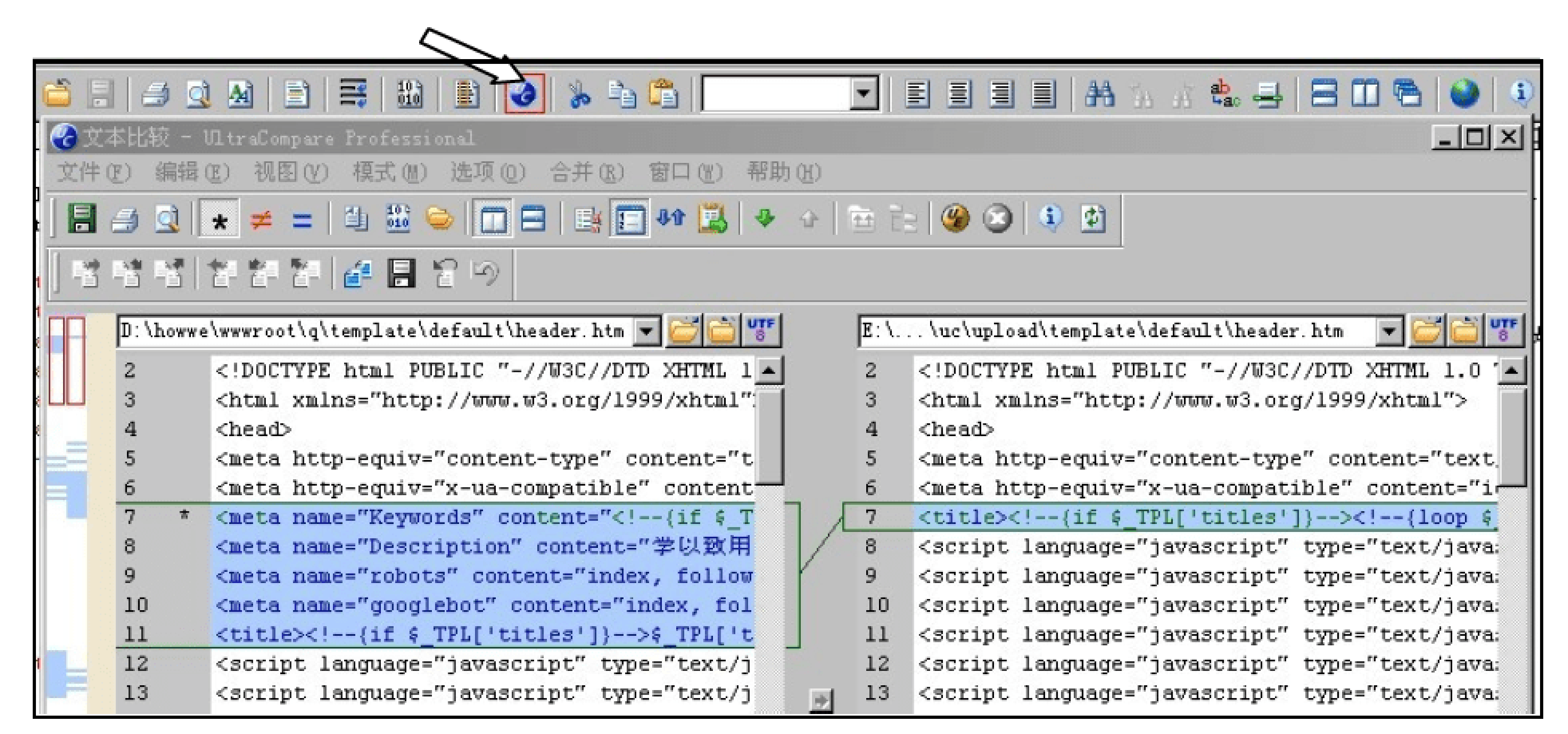


图 13-28　使用 UltraEdit 比较 header.htm

注意：

当 UltraEdit 打开的文件为两个时，不用选择比较的文件，可直接进行对比。否则必须选择文件，如图 13-29 所示，已打开三个文件，所以在“比较文件”时需选择待比较的文件。

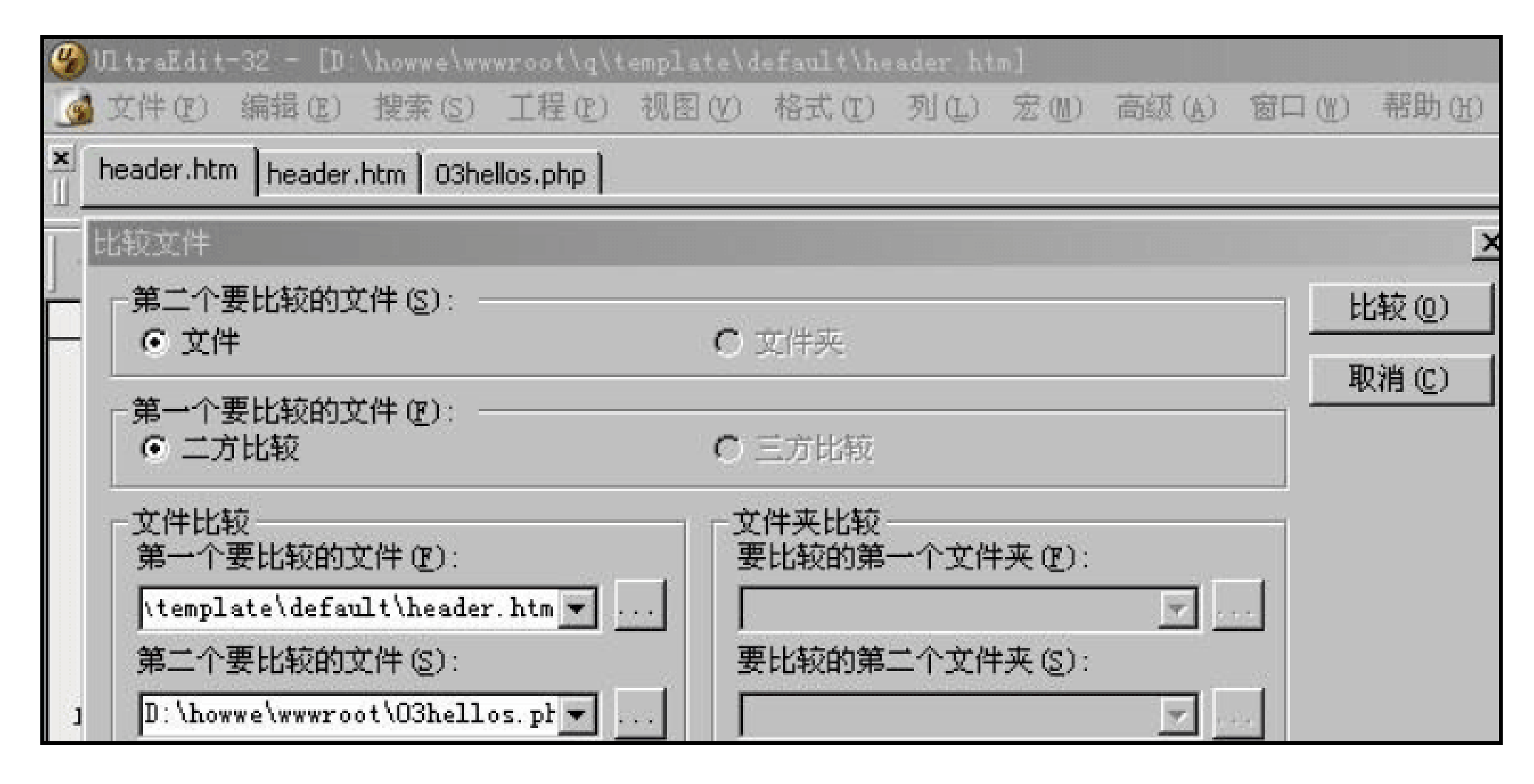


图 13-29　UltraEdit 比较文件前需选择

5. 页脚文件 footer.htm 的修改

页脚的修改比较简单，修改不大，原始版和修改版的页面截图分别如图 13-30 和图 13-31 所示。

我的空间 - 联系我们

Powered by **UCenter Home** 2.0 © 2001-2009 Comsenz Inc.

图 13-30　UCenter Home 标准页脚

学以致用 对接需求-浩为资源堂 浩为IT培训，资源整合 - 联系我们 - 沪ICP备09068053号
上海浩为通信技术有限公司,基于**UCenter Home**.

图 13-31 浩为资源堂的页脚

修改版修改部分的代码如下，不再详述，位置在第 35~42 行：

```
<p>
    <a href="http://howwe.net">学以致用 对接需求-浩为资源堂 浩为 IT
      培训，资源整合</a> -
<a href="mailto:$ SCONFIG[adminemail]">联系我们</a>
    <!--{if $ SCONFIG['miibeian']}--> - <a  href=
      "http://www.miibeian.gov.cn" target=" blank">$ SCONFIG
      [miibeian]</a><!--{/if}-->
</p>
<p>
    上海浩为通信技术有限公司,基于<strong>UCenter Home</strong>.
</p>
```

6. 日志的修改

日志部分的修改不大，只是在页面中增加了与“群组”相关的内容。为了简化 URL，以 space.php 为基础，整合相关代码，形成单独的文件 me.php，位置 D:\howwe\wwwroot\q，详细代码请自己去浏览，在 UltraEdit 中显示为 520 行。

用户名和密码均为 admin，登录 http://127.0.0.1/q，单击左侧菜单栏的“日志”，将打开链接 http://127.0.0.1/q/me.php?view=all&orderby=dateline，显示内容如图 13-32 所示，显示内容为“浩为资源堂‘日志’列表”。在图 13-32 的右下角，有“群组：谈谈灌灌”的字样，表明日志“资源整合才能使中国动漫业焕发活力”属于群组“谈谈灌灌”。但是当日志没有选择所属群组时，不会显示“群组”信息，如图 13-33 所示。



图 13-32 浩为资源堂的“日志”列表

java程序员的感想　分享 +
邹丽梅 2010-08-27 12:55

从刚刚开始接触软件开发，觉得自己挺了不起的，因为很多东东在那些刚刚要接触软件开发的人来说都很难，但是我都一一排解了，等接触了软件开发半年多之后就知道中心里教的东东本来就是比较肤浅的，那以我的智力和脑力可以做到那样子确实是理所应当的，到了接触一年多之后就觉得老师虽然挺强的，但是他们知

3 次阅读 | 2 个评论

图 13-33　“日志”未指定所属“群组”

代码分析

1) URL 分析

先看访问的 URL——http://127.0.0.1/q/me.php?view=all&orderby=dateline，再列出该页中和 me.php 相关的 URL：

好友最新日志——http://127.0.0.1/q/me.php?uid=1&view=we
大家的日志——http://127.0.0.1/q/me.php?uid=1&view=all
我的日志——http://127.0.0.1/q/me.php?uid=1&view=me
我表态过的日志——http://127.0.0.1/q/me.php?uid=1&view=click

推荐阅读——http://127.0.0.1/q/me.php?view=all
最新发表——http://127.0.0.1/q/me.php?view=all&orderby=dateline
人气排行——http://127.0.0.1/q/me.php?view=all&orderby=hot&day=7
评论排行——http://127.0.0.1/q/me.php?view=all&orderby=replynum&day=7
查看排行——http://127.0.0.1/q/me.php?view=all&orderby=viewnum&day=7
路过排行——http://127.0.0.1/q/me.php?view=all&orderby=click_1&day=7
鸡蛋排行——http://127.0.0.1/q/me.php?view=all&orderby=click_5&day=7
鲜花排行——http://127.0.0.1/q/me.php?view=all&orderby=click_4&day=7
握手排行——http://127.0.0.1/q/me.php?view=all&orderby=click_3&day=7
雷人排行——http://127.0.0.1/q/me.php?view=all&orderby=click_2&day=7

资源整合才能使中国动漫业焕发活力——http://127.0.0.1/q/me.php?uid=2&id=571
专业的网站策划方案书写法、演示标准——http://127.0.0.1/q/me.php?uid=2&id=570
FCKeditor 在 Firefox 下删除段前空格——http://127.0.0.1/q/me.php?uid=2&id=569
… …

分析以上 URL，可以看出：有参数 view 时，显示的内容为日志列表；有参数 id 时，显示日志的具体内容。

2) me.php 分析

如图 13-34 所示，显示了 me.php 的部分代码及其相关文件。

为了处理诸如 http://127.0.0.1/q/me.php?62(资源堂必读)中的“?62”，使用图 13-34 中的第 8~13 行代码，对 QUERY_STRING 做特别处理，如果 QUERY_STRING 为数字，则默认其值为 id 的值。

UltraEdit-32 - [D:\howwe\wwwroot\q\me.php]

文件(F) 编辑(E) 搜索(S) 工程(P) 视图(V) 格式(T) 列(L) 宏(M) 高级(A) 窗口(W)

me.php | common.php | data_magic.php | space_blog_list.htm | space_blog_view.htm

```
1  <?php
2  include_once('./common.php');
3  include_once(S_ROOT.'./data/data_magic.php');
4
5  //是否关闭站点
6  checkclose();
7
8  if(is_numeric($_SERVER['QUERY_STRING'])) {
9     $id = intval($_SERVER['QUERY_STRING']);
10 }
11 else{
12    $id = empty($_GET['id'])?0:intval($_GET['id']);
13 }
14
15 //获取变量
```

图 13-34 me.php 部分代码及相关文件

me.php 代码概述：

- 2-3 行：引入包含文件。
- 8-13 行：获取$id 的值。若?后的内容为数字，则赋值给$id，否则读取参数 id 的值。
- 15-35 行：主要是获取参数。注意点：已指定$do = 'blog';。
- 37-42 行：读取“群组分类”，并将其值赋给$tagarr[$value['tagid']]。
- 44-55 行：如$id 有值，则读取其内容给$blog。
- 58-157 行：UC Home 通用功能的实现。
- 159-520 行：具体功能的实现。
- 160-320 行：显示具体日志内容，include_once template("space_blog_view")包含模板 space_blog_view。
- 322-520 行：显示日志列表，include_once template("space_blog_list");包含模板 space_blog_list。

3) 日志列表 space_blog_list 对“群组”的显示

打开 D:\howwe\wwwroot\q\template\default\space_blog_list.htm，查找“群组”，在第 109 行有以下内容：

```
    <!--{if $tagarr[$value[tagid]]}--><span class="pipe">|</span>群组: <a
href="q.php?$value[tagid]">{$tagarr[$value[tagid]]}</a><!--{/if}-->
```

其中$value[]为日志列表$list[]的一条日志，如果$tagarr[$value[tagid]]的值存在，就显示具体的群组信息。

注意：

$tagarr 必须先在 me.php 中定义。

4) 日志内容 space_blog_view 对“群组”的显示

在浏览器中打开日志“资源整合才能使中国动漫业焕发活力”，部分内容如图 13-35 所示，显示有“群组：谈谈灌灌”(URL 为 http://127.0.0.1/q/me.php?uid=2&id=571)。



图 13-35 浩为资源堂日志

打开 D:\howwe\wwwroot\q\template\default\space_blog_view.htm，查找“群组”，在第 45 行有以下内容：

```
<!--{if $tagarr[$blog[tagid]]}-->群组:
    <a href="q.php?$blog[tagid]">{$tagarr[$blog[tagid]]}</a>
    <span class="pipe">|</span>
    <!--{/if}-->
```

其中$ blog []为日志的具体信息，如果$tagarr[$blog[tagid]]的值存在，就显示具体的群组信息。

注意：

$tagarr 必须先在 me.php 中定义。

5) 日志的修改

日志打开后，如果登录用户有权修改该日志，则在日志正文的右下方有一“编辑”链接，如单击日志“资源整合才能使中国动漫业焕发活力”的“编辑”后，将出现图 13-36 所示的日志编辑界面，URL 为 http://127.0.0.1/q/cp.php?ac=blog&blogid=571&op=edit。

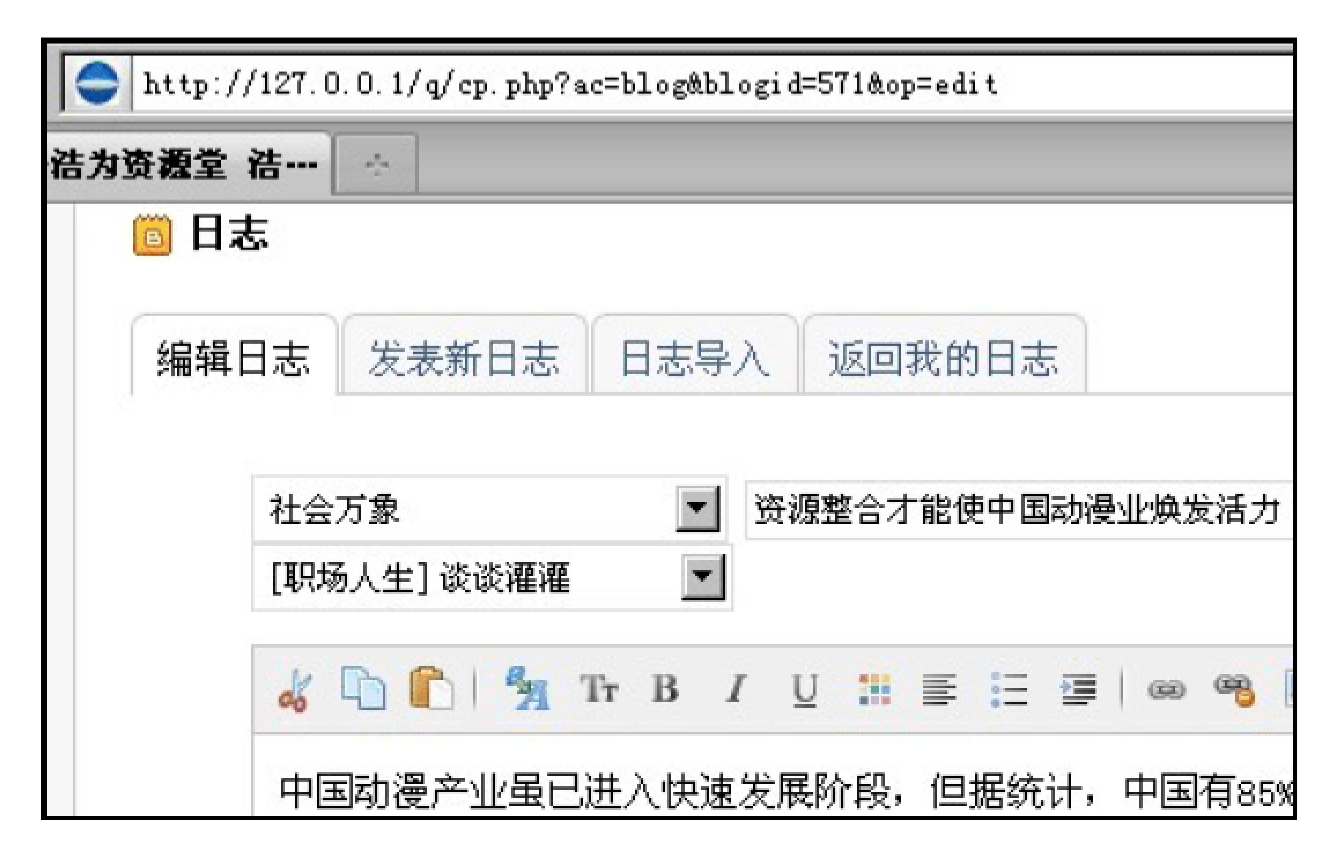


图 13-36　编辑日志

打开 cp.php，因为$ac 为blog，include_once(S_ROOT.'./source/cp_'.$ac.'.php');用来包含文件 cp_blog.php。cp_blog.php 在 D:\howwe\wwwroot\q\source 下，打开后相关代码如图 13-37 所示，代码为第 132～199 行，模板文件为 cp_blog.htm。

```
UltraEdit-32 - [D:\howwe\wwwroot\q\source\cp_blog.php]
文件(F) 编辑(E) 搜索(S) 工程(P) 视图(V) 格式(T) 列(L) 宏(M) 高级(A) 窗口(W) 帮助(H)
cp.php  cp_blog.php
132 } else {
133   //添加编辑
134   //获取个人分类
135   $classarr = $blog['uid']?getclassarr($blog['uid']):getclassarr($_SGLOBAL['s
136
137   //2009.11.18 ayu add
138   include_once(S_ROOT.'./data/data_profield.php');
139   //我的群组列表
140   $mtaglist = array();
141   $query = $_SGLOBAL['db']->query("SELECT main.*,field.tagname,field.membernu
142     LEFT JOIN ".tname('mtag')." field ON field.tagid=main.tagid
143     WHERE main.uid='$_SGLOBAL[supe_uid]' AND main.grade>=0");
144   $havemtag = false;
145   while ($value = $_SGLOBAL['db']->fetch_array($query)) {
146     $havemtag = true;
147     if(empty($value['close']) && $value['membernum']>=$_SGLOBAL['profield'][$
148       $mtaglist[$value['fieldid']][$value['tagid']] = $value;
149     }
150   }
151   //->
```

图 13-37　日志编辑处理代码

在图 13-37 所示的代码中，第 137～151 行用于定义群组变量$mtaglist[]。

打开 cp_blog.htm，文件位于目录 D:\howwe\wwwroot\q\template\default 下，查找“群组”，在第 85～96 行有以下内容：

```
              <select name="tagid" id="tagid">
               <option value="0">可选择将发布的群组</option>
          <!--{loop $mtaglist $fieldid $values}-->
          <!--{loop $values $value}-->
          <!--{if $value[tagid] == $blog['tagid'] }-->
```

```
            <option value="$value[tagid]"
               selected>[{$_SGLOBAL[profield][$value[fieldid]]
               [title]}] $value[tagname]</option>
        <!--{else}-->
            <option value="$value[tagid]">[{$_SGLOBAL[profield]
             [$value[fieldid]][title]}] $value[tagname]</option>
        <!--{/if}-->
        <!--{/loop}-->
        <!--{/loop}-->
        </select>
```

该代码用于显示群组信息，详情请自行阅读。

6) 发表新日志

单击左侧菜单栏“日志”后的“发表”，出现图 13-38 所示的界面，其中有一下拉框为“可选择将发布的群组”，该内容的显示同“**5)日志的修改**”，具体不再说明。

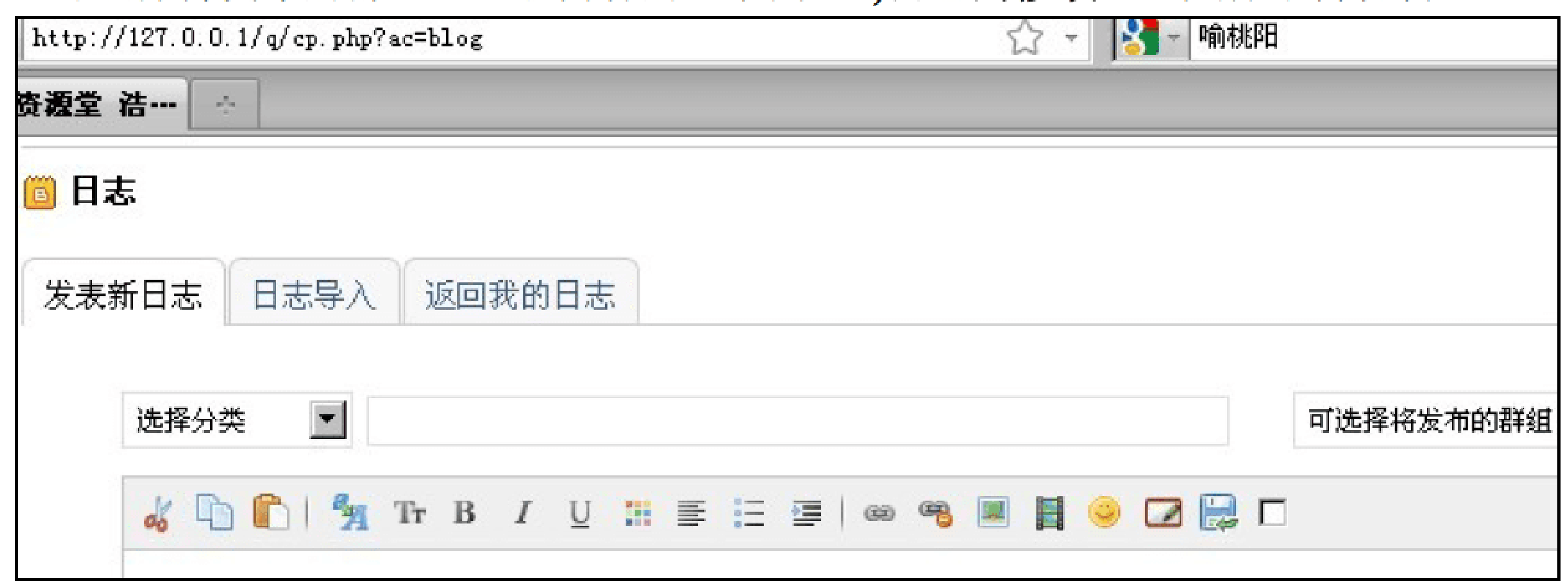


图 13-38 发表新日志

唯一要说明的是：单击“保存发布”时，起作用的参数为 blogsubmit，在 cp_blog.htm 的第 188 行，处理文件为 cp_blog.php，代码为 58-84 行。在这些代码中，又包含了另一个文件：source/function_blog.php，使用其中的 blog_post 对新日志或日志更新进行处理，在 function_blog.php 的 120 行，`'tagid' => intval($tagid)`用于处理群组信息。更多内容可搜索该文件中的“tagid”。

7. 群组的修改

群组的修改涉及的代码很多，主要由文件 ok.php 及 q.php 完成。但由于原有代码的混乱，其修改没有按照最初的设想去改，如对话题的修改，改了几次代码还是有问题，以致无心再改，只得变通变通，直接调用日志的修改界面。为啥不想再改呢？因为那几个文件的逻辑不太清楚，除非心特别平静，否则那些代码只会越看越烦，以致不想再看。为啥会越看越烦呢？就在于代码的乱，而乱的原因就在于没有整体思路。所以在写代码时，一定要注意代码的整体性，尤其在团队合作时，更应注意代码的整体性。没有整体性，自己写

代码很累，别人看代码更累；以后的修改更是无从下手。

1) 在群组中发表话题

单击左侧菜单栏“群组”后的“话题”，出现图 13-39 所示的界面，第一个下拉框为群组列表，第一个群组为默认群组，第二个下拉框为“日志位置”，可从下拉框中选择该日志的分类，如不选择，则不属于任何日志分类。

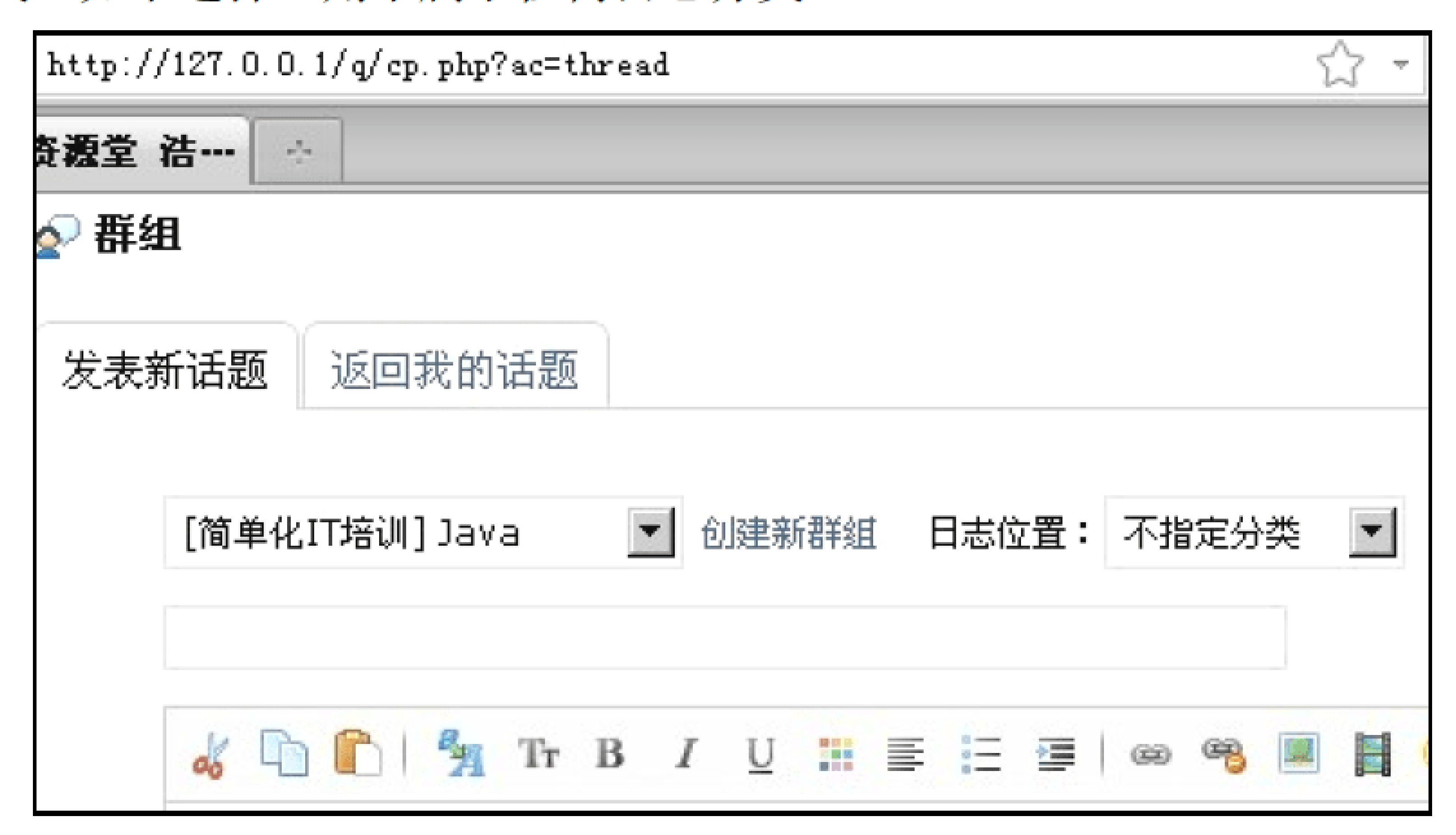


图 13-39 在群组中发表话题

打开 cp.php，$ac 为 thread，include_once(S_ROOT.'./source/cp_'.$ac.'.php');包含文件为 cp_thread.php。cp_thread.php 在 D:\howwe\wwwroot\q\source 下，模板文件为 cp_thread.htm。

注意：

cp_thread_Old.php 为原始版本的 cp_thread.php，可对比两种版本间的区别。

打开 cp_thread.htm，查找“日志位置”，在 202～312 行有如下代码：

```
    <!--{if !$tagid or !$thread}-->
  日志位置：<select name="classid" id="classid" onchange="addSort(this)">
                <option value="0">不指定分类</option>
                <!--{loop $classarr $value}-->
                <!--{if $value['classid'] == $blog['classid']}-->
                <option value="$value[classid]" selected>
                    $value[classname]</option>
                <!--{else}-->
                <option value="$value[classid]">
                  $value[classname]</option>
                <!--{/if}-->
                <!--{/loop}-->
            </select>
    <!--{/if}-->
```

该代码用于显示日志分类信息，详情请自行阅读。注意$classarr 用于日志分类，定义在 cp_thread.php 的第 631～633 行。

单击“保存发布”时，起作用的参数为 threadsubmit，在 cp_thread.htm 的第 257 行，处理文件为 cp_thread.php，代码为 34～253 行。也可打开 cp_thread_Old.php 对比对比，体会一下为何要那样修改。

2) 修改话题

正如前面所说，对话题的修改是直接使用日志的修改。请参考前面相关内容，不再讲述，但也可以去看看原版是如何修改话题的。

对比 cp_thread.php 和 cp_blog.php，可以看到，后者代码的可读性很强，具体请自己去体会吧！

3) ok.php 分析

如图 13-40 所示，显示了 ok.php 的部分代码及其相关文件。

为了处理诸如 http://127.0.0.1/q/ok.php?62(资源堂必读)中的“?62”，可以使用图 13-40 中的第 8～13 行代码，对 QUERY_STRING 做特别处理。如果 QUERY_STRING 为数字，则默认其值为 id 的值。

UltraEdit-32 - [D:\howwe\wwwroot\q\ok.php]
文件(F)　编辑(E)　搜索(S)　工程(P)　视图(V)　格式(T)　列(L)　宏(M)　高级(A)
ok.php | space_thread_list.htm | space_thread_view.htm

```
1  <?php
2  include_once('./common.php');
3  include_once(S_ROOT.'./data/data_magic.php');
4
5  //是否关闭站点
6  checkclose();
7
8  if(is_numeric($_SERVER['QUERY_STRING'])) {
9      $id = intval($_SERVER['QUERY_STRING']);
10 }
11 else{
12     $id = empty($_GET['id'])?0:intval($_GET['id']);
13 }
14
15 //获取变量
```

图 13-40　ok.php 部分代码及相关文件

ok.php 代码概述：

- 2-3 行：引入包含文件。
- 8-13 行：获取$id 的值。若?后的内容为数字，则赋值给$id，否则读取参数 id 的值。
- 15-45 行：主要是获取参数。注意点：已指定$do = 'thread';。
- 47-58 行：如$id 有值，则读取其内容给$thread。
- 59-153 行：UC Home 通用功能的实现。
- 155-357 行：具体功能的实现。

- 155-273 行：显示具体话题内容，include_once template("space_thread_view")包含模板 space_thread_view，将从 q_thread 与 q_post 读取数据改为从 q_blog、q_blogfield、q_comment 读取数据。
- 276-357 行：显示群组话题列表，include_once template("space_thread_list");包含模板 space_thread_list，将从 q_thread 读取数据改为从 q_blog 读取数据，群组话题列表界面如图 13-41 所示。



图 13-41 群组话题列表

4) q.php 的修改

q.php 主要用于显示群组信息，修改不大，只指定了$do = 'mtag';，在第 19 行，界面如图 13-42 所示，修改主要是将从表 q_thread 读取数据改为从 q_blog 读取数据。



图 13-42 群组信息

8. 后记

本章对代码的讲解较少，因为不需要靠讲解代码来拉长篇幅，主要介绍如何分析并解决问题，即只提供整体思路。具体代码请在掌握整体思路的前提下阅读。只有这样，在整体思想的指导下多阅读代码，你的开发水平才会迅速提高。

第 14 章

PHP & Java 的整合

PHP 简单易学，开发动态网页效率高，但草根文化气息浓重，是语言世界的平民英雄。

Java 强大而复杂，有众多高端功能，更有 IMB、Oracle 等企业大鳄的追捧，是语言世界的贵族王子。

不管是平民英雄还是贵族王子，都不满足在各自领域称王称霸。于是，平民英雄祭出 LAMP 大旗，并在 PHP 5 中增加对象模型，试图往上一阶层硬闯，可是金融、电信等应用王侯不大欢迎；而贵族王子也想重视 Web 基层社会，先制定 Servlet/JSP，接着推出 struts、JSF、Taglib、JSTL、Spring 等系列框架，试图吸引更多的平民百姓，但因为其实施手法复杂而无法深得民心。

如何才能充分利用两者的优势，PHP 结合 Java 是很多人在思考的问题：PHP 主要负责 Web 层，而 Java 负责业务和数据逻辑层，形成一对黄金组合。目前已有多种解决方案：1) PHP 中内置调用 Java 的方法(使用 PHP 的 Java 扩展模块)；2) 使用 minij2ee 应用服务器提供的 SJOP(Sample Java ORB Protocol，简单 Java 对象请求代理协议)协议实现；3) 采用 PHP/Java Bridge；4) 使用 Quercus；5) 通过 SOAP(Simple Object Access Protocol)。

本章重点介绍基于 Quercus 的应用。

14.1　PHP & Java 整合的必要性

以前提到 Java 的最大特点是跨平台、一次编写到处运行。近几年来，Java 领域最大的变化是基于 JVM 的语言开始流行，Java 已经进入了混合编程时代。Java 和 PHP 的整合也是这一趋势的表现。

在 Web 开发市场中，Java 和 PHP 都是目前应用的热门技术。Java 的强大不容置疑，不仅体现在 WEB 的开发上，在各个软件应用领域，Java 也无所不在。而 PHP 是全球 2200 万个网站都在使用的语言，可见 PHP 的发展趋势不可小觑，再加上 PHP 有开源力量及

ZEND、IBM、Oracle 等公司的大力推动，PHP 日渐繁荣。有人预言“PHP 将比 Java 更受欢迎”，但 PHP 永远也不可能完全替换 Java，两种的融合更是技术发展的趋势。

两种技术可以激烈竞争，也可以紧密合作。从客户的角度来说，技术之间的融合非常重要，因为一个庞大的应用系统，通常不可能由单一语言或技术独立完成。PHP 和 Java 的融合，为需要结合 PHP 与 Java 的企业提供了一个良好的选择。

1. PHP 为什么需要 Java

1) PHP 中的组件都是短暂、非持久性的。对于复杂的应用体系，必须提供中间层组件(如 Java beans/EJB)或者企业级的缓存技术、连接池或商业逻辑给 PHP 组件生成页面。例如，解析 XML 文件是一个比较耗资源的工作，需要缓存；连接数据库更是一个比较耗资源的工作，需要资源重用。标准的 PHP XML 和 DB 抽象层效率很低，因为它们都不能通过一个中间层来实现缓存和连接池。

2) 即使是一些小任务，也可能需要用到 Java Class 或 Java 类库，例如需要跨平台地生成 Word、Excel 或 PDF 文档。

3) PHP 代码、PHP/Java Bridge 可以打包成标准的 J2EE 档案包格式，以方便布置到一个 J2EE 应用服务器或 Servlet 引擎中，而用户不需要安装 PHP。从用户的角度来说，用户看不到这些用 JSP、Servlet 或 PHP 生成的页面有什么区别。由于 Bridge 允许 PHP 和 J2EE 间的 session 共享，所以开发者可以一步步地把基于 JSP 的程序和 PHP 集成起来。

2. Java 也存在不足

对于 Java 程序员来说，PHP 和 PHP/Java Bridge 也许可能有用。许多基于 JSP 的技术，如 Struts 及 JSF 存在缺陷，再把它们整合在一起去建立面向对象的 WEB 系统，就更暴露了这些问题。即使 JSF 的作者也承认了这样的系统有严重缺陷，并推荐用像 tapestry 或 facelets 等用 Java 类定义的组件并通过它们的 ID 绑定到 XML/HTML 模板中。PHP 与 Java 的整合可以将 PHP 代码嵌入到 Java 系统中，这样用户界面设计师就可以集中精力设计 HTML 模板，而程序员可以用 PHP 建立原型，并使用已有的开发框架。现在，不少大型站点前端使用 PHP，而核心使用 Java 等语言来构建。

3. JSR 及 JSR223

JSR(Java 规范请求)是指向 JCP(Java Community Process)提出新增一个标准化技术规范的正式请求。任何人都可以提交 JSR(Java 规范请求)，以向 Java 平台增添新的 API 和服务。JSR 已成为 Java 界的一个重要标准。

在 Java Servlet 规范(Servlet 2.4，JSR-154)中，定义了一系列核心的抽象概念(WEB 程序处理过程中需要考虑的各种对象)来让 Java 程序员编写 WEB 程序，包括 Session、Request、Response 等。当程序员在编写程序时，可以很方便安全地与这些对象进行通信。JSR223 描述的是这些 Java 对象如何向用其他脚本语言编写的 WEB 页面开放，使其他语言也可以

访问这些对象。这个规范已被用于 PHP 及其他脚本语言，因为 JSR223 提出的概念独立于脚本语言，其目的是将脚本语言集成到 Java 平台之上。

14.2　PHP & Java 整合方案简介

1. PHP 的 Java 模块

PHP 发布版中包含一个 Java 扩展模块，可以用来调用 Java 对象，如：

```
<?php $system=new Java("java.lang.System");
 print "Java version=".$system->getProperty("java.version")." <br>";
?>
```

其优点是比较方便，只要用 new Java()就能创建一个 Java 对象，可以同 PHP 类那样调用 Java 对象。但这种方法有明显的缺点：

1) 由于 PHP 的 Java 模块根据 PHP 的数据类型选择最适合的 Java 方法，因此无法调用 Java 已加载的函数。

2) PHP 的 Java 模块将在当前 Web Server 进程中载入 JVM(Java 虚拟机)，因此系统开销极大，影响 Web Server 进程的执行效率。

3) PHP 的 Java 模块有时会使 Web Server 进程僵死。

由于以上原因，PHP 的 Java 模块一直无法应用到实际的软件系统中。

2. minij2ee 应用服务器 SJOP 协议实现

minij2ee 应用服务器是第一款支持 PHP 的 J2EE 应用服务器产品，使 PHP 能够用于开发企业级应用系统。SJOP 全称是 Sample Java ORB Protocol(简单 Java 对象请求代理协议)，是一种简单高效的对象请求代理协议。比如：

```
<?php
$conn=minij2ee_fetch_connection();
print  "Java version = ".minij2ee_callstatic_javaobj ($conn,
"java.lang.System", "getProperty", "java.lang.String", "java.version")."
<br>";
?>
```

minij2ee 应用服务器实现 SJOP 协议的主要目的是使 PHP 能够访问 EJB 企业级组件，因此 minij2ee 提供了一个 EJB-PHP 编译器，可以把 EJB 组件编译成 php 的类，使 php 程序能够方便地调用 EJB 组件，例如：

```
<?php
```

```
require("Cart.php"); //Cart.php 是编译 Cart EJB 后生成的 Cart EJB 的 php 类定义。
$home=new CartHome(); //创建 EJB 的 Home 接口。
$objref=$home->create($cart_name); //创建 Cart EJB。
$cart=new Cart($objref);
$cart->add("some goods");//向购物车中添加一个物品。
?>
```

使用 minij2ee 应用服务器的 PHP 支持，就可以开发出基于 PHP 和 J2EE 技术的、面向对象的、稳定高效的企业级应用系统。

官方网址为 http://www.minij2ee.com，该地址已无效，但该技术为第一款支持 PHP 的 J2EE 应用服务器产品，值得大家稍微了解。

3. PHP/Java Bridge

PHP/Java Bridge 包含一个 PHP 模块(包括 java.so、php_java.dll)和一个相关的后端程序(JavaBridge.jar、JavaBridge.war 或 MonoBridge.exe)，用于连接 PHP 的对象体系到 Java 或 ECMA 335(CLI，Microsoft .NET Framework 的重要子集)虚拟机。它完全实现了 JSR 223 规范请求，可以使 PHP 脚本访问基于 CLR(如 VB.NET、C#)或 Java(Java、KAWA、JRuby)的应用程序。PHP/Java Bridge 通过本地 socket 用一个高效的通信协议与虚拟机进行通信。一个多进程的 HTTP 服务器的每个处理请求的 PHP 进程都会有一个相应的虚拟机进程。

多个 HTTP 服务器的请求会被集中发送到一个运行着 PHP/Java Bridge 的应用服务器，或者每个 HTTP 服务器都有一个 PHP/Java Bridge 来和一个 J2EE 应用服务器进行通信，必需的客户端类库(EJB client.jar)将在运行时被加载。

如果在 ECMA 虚拟机内至少有一个后端程序在运行，如 Novell 的 MONO 或.NET，那么基于 ECMA 335 的类就可以被访问，并可以支持 varargs、reflection、assembly loading 等特性。如果在 J2EE 环境中有一个后端程序运行，PHP 和 JSP 间就可以实现 session 共享，更可以实现集群和负载平衡。

网址为 http://php-java-bridge.sourceforge.net。

4. SOAP(Simple Object Access Protocol)

SOAP 是 IBM、Microsoft 等公司开发、W3C 推荐，用来实现分布式对象技术的协议。SOAP 提供了一套以 XML 来包装程序调用、参数传递与信息回传的机制，借助 XML 纯文字的特性，可通过 HTTP、HTTPS、SMTP 等通信管道穿越企业的防火墙。比起 CORBA、Java RMI 及 DCOM 这些以专属 binary 格式传送数据的分布式对象技术协议，SOAP 具有与程序语言、平台和硬件无关的特性。

Java 语言最常使用的 SOAP 套件是 Apache Axis2，PHP 使用的是 php-soap 延伸模块。下面说明如何通过这两个 SOAP 套件整合 PHP 与 Java。

前提：安装 PHP >=5.0 版、JDK >=1.4 版以及 Tomcat。

基本原理：使用 Apache Axis2 将 Java 部署成 SOAP 的 Web Services(如 HelloService)，再使用 php-soap 延伸模块让 PHP 调用 HelloService 服务。详情请自己去 Google。

小知识：LAJP 作者对 LAJP 的评论

LAJP 也是一种 PHP 与 Java 的整合方案，名字来源于 LAMP(Linux、Apache、MySQL、PHP)，LAMP 是轻量级的 Web 开发运行环境，在 Internet 上有广泛的应用，但对于企业开发，如金融、电信领域，LAMP 显得能力不足，这些领域通常是 Java(J2EE)的势力范围。LAJP 将 LAMP 的简便性和 Java 能力结合起来，LAJP 中的 J 指的是 Java。LAJP 的相关资料，可查看网站 http://code.google.com/p/lajp，为开源软件。图 14-1 为 LAJP 的基本构架。

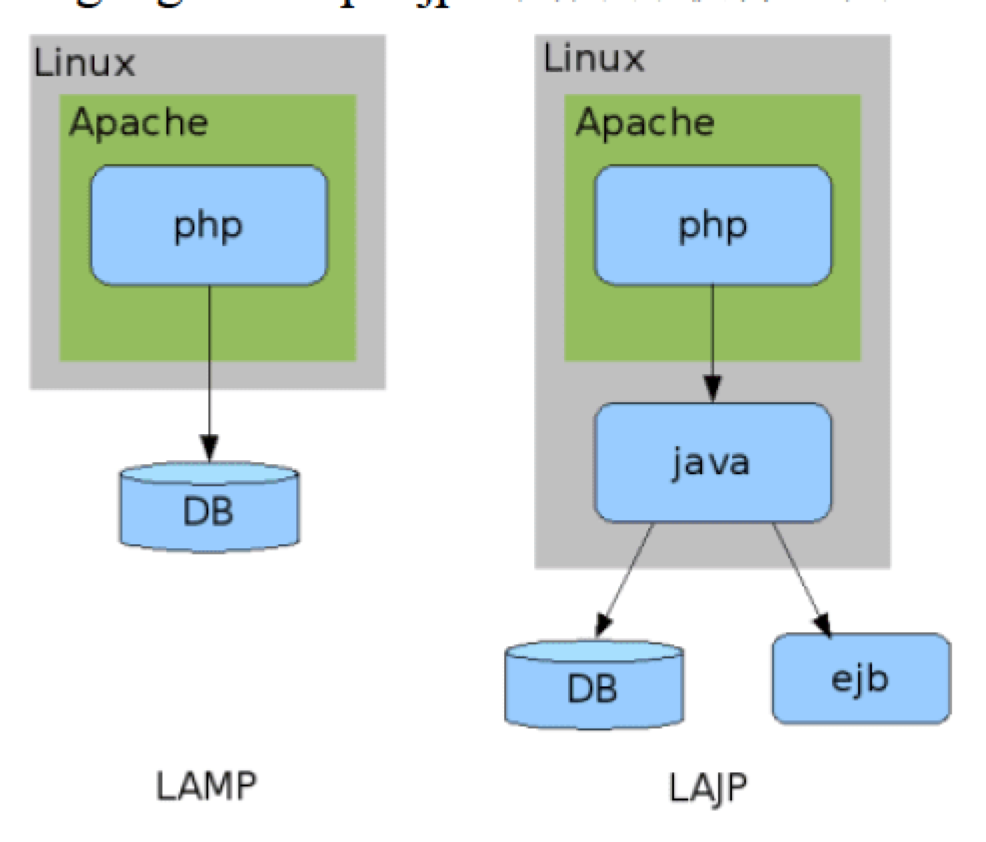


图 14-1 LAJP 的基本构架

我很赞同一个观点：“Web 开发人员不应对 PHP 或 Java 抱有成见，而应审时度势，结合 PHP 与 Java 各自的优势，才能更有效率地开发 Web 应用程序”。这是具有大局观、大境界的观点，看似简单，实则深奥。在创建 LAJP 这个开源项目时，我预想了可能遇到的阻力：

- 从结构上看 LAJP 和 JSP+JavaBean 是一样的，既然已有 JSP+JavaBean，为什么还要 LAJP？
- 我是 Java 程序员，LAJP 要求学习 PHP，我不愿意学，有困难。
- LAJP 是个人负责的开源软件，技术上有风险。

1) 不考虑底层实现，从架构上看，LAJP 和 JSP + JavaBean 确实一样，但换个角度来看，如果单纯用 JSP 或 PHP 开发 Web 系统，哪种更好？而 LAJP 将 JSP + JavaBean 转换成 PHP + Javabean，可以取长补短。

2) Java 程序员看不起、也不愿学 PHP，这是对 PHP 的成见。不少人以为 PHP、ASP 程序员水平不高，可现在流行的开源 Web 软件，PHP 版的质量明显高于 Java，如 Wordpress、phpBB 等软件，目前还没出现 Java 开发的能望其项背的软件。而且 PHP 相当易学，对一个从事过 Web 开发的 Java 程序员，PHP 的学习难度远小于 Struts、Spring，PHP 号称 5 分钟上手，资质差的一小时上手，两星期熟练不会有问题，况且 PHP 原生函数能带给你在操

作系统、图像编辑等新领域的知识，也不会让你为一个上传组件而找一堆 jar 文件，何乐而不为呢？

3) 虽然我自己很有信心，但未经过长时间、大应用的考验，LAJP 能否作为一个稳定可靠的架构组件有待考验。下面我讲讲 LAJP 使用的技术，以供大家分析：

先说 PHP 端，LAJP 不用要求添加.so，只添加了一个 php_java.php 文件，在这个文件里运用了三项技术：PHP 序列化，System V 消息队列和 System V 信号量。序列化是 PHP 语言的核心，System V 在 Unix 中已存在 20 多年，根本不用担心。

LAJP 的实现主要在 Java 端，大部分代码为 PHP 序列化数据的解析和组装，基本都是枯燥的字符串拼接操作，这里可能有小 bug，但不会有硬伤；其次，运用了 Java 的反射机制来查找类、方法，并调用执行方法，这些和 Struts、Spring 的实现是相似的(JDK 中提供了那几个反射操作)，也不大会出问题；第三，通过 JNI 操作 System V 信号量、消息队列，JNI 是比较容易出问题的地方，但我认为不大可能出现问题，因为其实现太简单了。

最后，也是我在 LAJP 上思考最多、且至今还没有更好改善的地方是多线程服务。在 LAJP 中，PHP 可看作是客户端，Java 是服务端，当 PHP 发起一个服务调用时，Java 必须应对一个服务线程，这称之为并发服务，Tomcat、Websphere 的服务也是这样。从稳定性上讲，并发线程相对于并发进程要差很多。在设计服务线程时，超时问题是不能回避的，但在多线程服务中，还没有完美的停止一个服务线程的方法，这不是 Java 放弃了线程 stop()方法造成的原因，而是多线程本身的机制。而 Java 的 JVM 机制，无法做成多进程并行机制(太耗内存)，这也是 Tomcat 相比 Apache 不稳定的重要因素之一。相对于使用 Tomcat、Websphere 这些纯 Java 的服务，LAJP 在理论上要稳定一些。但最可能出问题的是，HTTP 服务这一块交给了 Apache，而 LAJP 中 Java 服务超时的几率要小很多。

参考资料：

1) 线程相关的资料可阅读《UNIX 环境高级编程》“第 11 章　线程”和“第 12 章　线程控制”，在这里不建议阅读 Java 的书籍，因为 Java 线程“丢失”了太多的细节。

2) System V 消息队列、信号量相关的资料可阅读《UNIX 网络编程(第二卷)》的进程间通信部分。对于 System V 不利的小道消息是，这项技术老得连最初作者是谁都搞不太清，现在更是没人来维护；有利的消息是，有大量的服务长期在运用着它，例如 Oracle。

14.3　配置 Quercus

Quercus 是 Caucho 公司采用纯 Java 开发的一个 PHP 5 引擎(要求 JDK 1.5)，可以让 PHP 程序在 JVM 上执行。Quercus 自带很多 PHP 模块和扩展，如 PDF、PDO、MySQL 和 JSON 等。可以利用该引擎在 JavaEE 应用容器(Tomcat、Resin、GlassFish 等)中运行 PHP 程序，也可以在 PHP 脚本中调用 Java 服务，如 JMS 等。Quercus 预先将 PHP 文件编译成.java 文

件，再编译成 class，然后再执行。

Quercus 基于开源授权协议 GPL 发布，网址为 http://quercus.caucho.com。

1. 整个服务器配置 Quercus

下面讲解如何在 Howwe 开发包中为整个服务器配置 Quercus。(提示：开发包中的 Java 服务器为 Tomcat 的修改版 HoCAT，其他服务器的配置可依此类推)。

(1) 下载 quercus-4.0.8.war，http://caucho.com/download/quercus-4.0.8.war。

(2) 先将下载的 war 文件放在 D:\howwe\hocat\webapps，启动 D:\howwe\hocat 下的 hocat.exe，待 war 释放成一个目录 quercus-4.0.8 后，关闭 hocat.exe。

(3) 打开 D:\howwe\hocat\webapps\quercus-4.0.8\WEB-INF 下的 web.xml，在文件中有如下内容：

```
<servlet>
  <servlet-name>Quercus Servlet</servlet-name>
  <servlet-class>com.caucho.quercus.servlet.QuercusServlet
  </servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>Quercus Servlet</servlet-name>
  <url-pattern>*.php</url-pattern>
</servlet-mapping>
```

(4) 打开 D:\howwe\hocat\conf 下的 web.xml，先查找是否存在字符串“Quercus Servlet”，如存在，请删除对应的“<servlet>”和“<servlet-mapping>”，再将步骤 3 中的内容放在第一个“<servlet>”字符串的前一行，保存后如图 14-2 所示。

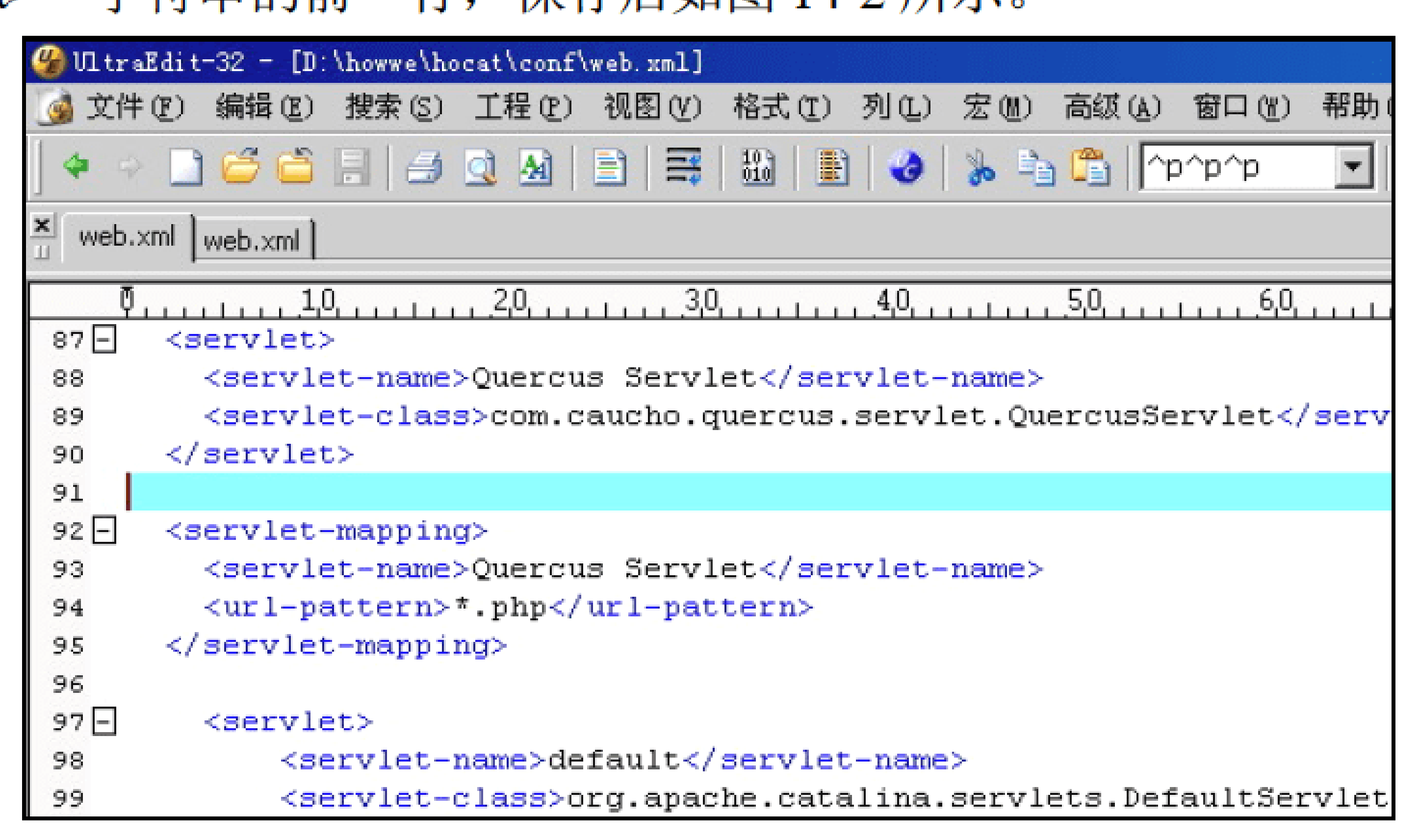


图 14-2　在 web.xml 中配置 Quercus Servlet

(5) 将 D:\howwe\hocat\webapps\quercus-4.0.8\WEB-INF\lib 下的三个 lib 文件：inject-16.jar、javamail-141.jar、resin.jar 复制到 D:\howwe\hocat\lib 下。

(6) 删除目录 D:\howwe\hocat\webapps\quercus-4.0.8。

重新启动 hocat.exe，Quercus 立即生效。

2. 单一项目配置 Quercus

如果只需要某个项目支持 PHP，直接修改 quercus-4.0.8 的目录名称即可。注意 war 文件也可以使用 winrar 解压，从本质上来说，.war 本身就是一种压缩格式。

注意：

Quercus 已经被集成在 Caucho 的 resin 中，可以放到 Tomcat 等 Java 容器中，据说和 Resin 的结合最好，因为两种为同一家公司的产品。

14.4 整 合 范 例

1. 在 D:\howwe\hocat\webapps\ROOT 下新建 1701.php

代码如下：

```
<?php
$l = new Java("java.util.ArrayList");
$l->add($buf=new Java("java.lang.StringBuffer"));
$buf->append("100");
echo ($l->get(0)->toString()) + 2;
echo '<br />';

// get instance of Java class java.lang.System in PHP
$system = new Java('java.lang.System');
// demonstrate property access
echo 'Java version=' . $system->getProperty('java.version') . '<br />';

// java.util.Date example
$formatter = new Java('java.text.SimpleDateFormat',
                 "EEEE, MMMM dd, yyyy 'at' h:mm:ss a zzzz");
echo $formatter->format(new Java('java.util.Date'));
?>
```

在浏览器中运行 http://127.0.0.1/1701.php，页面如 14-3 所示。

注意：

运行前必须先运行 D:\howwe\hocat 下的 hocat.exe，如运行时提示有错，请先关闭 PHP

运行环境，即运行 D:\howwe\php 下的 39-php-Stop.bat。

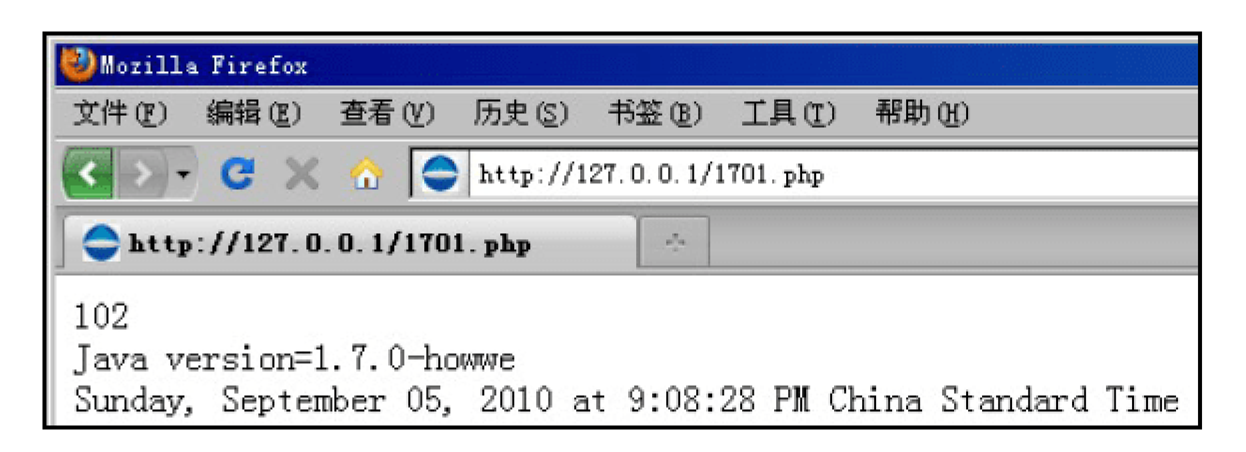


图 14-3　运行 1701.php

2. 在 D:\howwe\hocat\webapps\ROOT 下新建 1702.php

代码如下：

```
<?php
 echo "现在时间:\r";
 $a = new Java("java.util.Date");
 echo $a->time;

 $formatter = new Java('java.text.SimpleDateFormat',"yyyy-MM-dd
                        hh:mm:ss");
 echo "<br>格式化时间: ".$formatter->format(new Java('java.util.Date'));
?>
```

在浏览器中运行 http://127.0.0.1/1702.php，页面如图 14-4 所示。

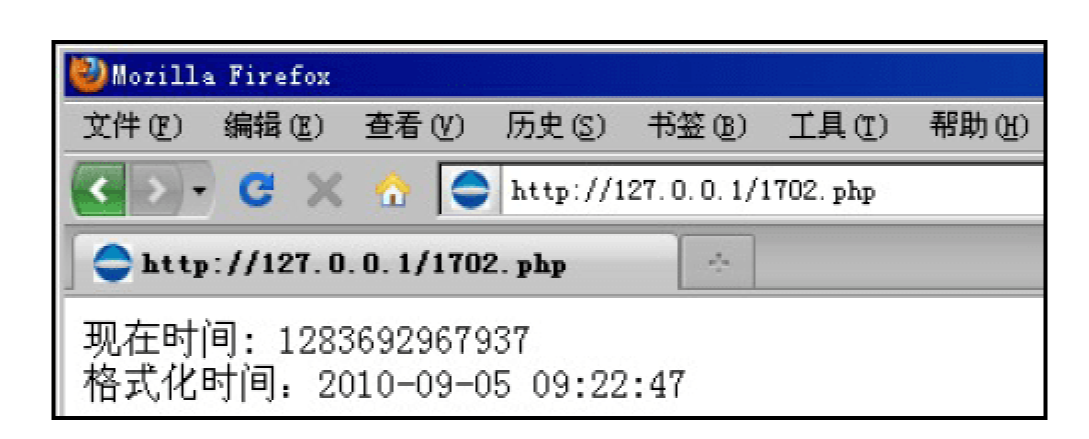


图 14-4　运行 1702.php

3. 在 D:\howwe\hocat\webapps\ROOT 下新建 1703.php

代码如下：

```
<?php
//Import classes
import java.util.Date;

$a = new Date(123);
print $a->time." -1<br>";
```

```
//Call Java methods
print $a->getTime()." -2<br>";
print $a->setTime(456)." -3<br>";

print $a->time." -4<br>";
?>
```

在浏览器中运行 http://127.0.0.1/1703.php，页面输出内容如下：

```
123 -1
123 -2
-3
456 -4
```

4. 在 D:\howwe\hocat\webapps\ROOT 下新建 1704.php

代码如下：

```
<?php
    $file = new Java("java.io.File", __FILE__, FALSE);
    var_dump($file);
    var_dump($file->isDirectory());
?>
```

在浏览器中运行 http://127.0.0.1/1704.php，页面输出内容如下：

```
resource(D:\howwe\hocat\webapps\ROOT\1704.php) bool(false)
```

此示例显示了使用内置 Java 类的 PHP 脚本，脚本将其存储在名为$file 的 PHP 变量中。然后脚本像处理普通 PHP 对象一样对该对象调用方法，如 isDirectory 方法。

此功能非常强大，允许 PHP 脚本访问任何 Java 类。请注意，该 Java 类必须位于应用程序类路径上，java.io.File 是核心 Java 类库的一部分，因此始终可用。

5. 在 D:\howwe\hocat\webapps\ROOT 下新建 1705.php

代码如下：

```
<?php
    $map = new Java("java.util.HashMap");

    $map->put("title", "Quercus!");
    $array = array(1, 2, 3, 4, 5);
    $map->put("stuff", $array);
    var_dump($map->get("stuff"));
    echo $map->get("title");
```

```
?>
```

代码分析：

1）创建 Java HashMap 类的实例。

2）将包含“Quercus！”的字符串存储在映射中。

3）突出 Java 和 PHP 类型之间的互操作性。

4）创建 PHP 数组，并将其存储在 Java 映射中，如下面的代码所示：

```
$array = array(1, 2, 3, 4, 5);
$map->put("stuff", $array);
```

对映射进行 put 调用时，PHP 数组会转换为最接近的 Java 类型，即 Java Map。与此类似，当 get 调用从$map 读取值时，会将其转换回常规的 PHP 数组。这可以在不进行任何复制的情况下进行，因为 PHP 数组具有两个个性类型，即 PHP 数组和 Java 映射。

在浏览器中运行 http://127.0.0.1/1705.php，页面输出内容如下：

```
    array(5) { [0]=> int(1) [1]=> int(2) [2]=> int(3) [3]=> int(4) [4]=> int(5) }
Quercus!
```

6. 在 D:\howwe\hocat\webapps\ROOT 下新建 1706.php

代码如下：

```
<?php
$list = new Java("java.util.ArrayList");
var_dump($list);
$date = new Java("java.util.Date", 70, 9, 4);
echo "<br/>";

$list->add("Quercus!");
$list->add($date);
$list->add(array(1, 2, 3, 4, 5));

$iterator = $list->iterator();
while ($iterator->hasNext() == TRUE) {
    var_dump($iterator->next()); echo "<br/>";
}
?>
```

代码说明：此例显示使用了 Java ArrayList 类，而且从 ArrayList 获得了迭代器，并从头到尾对集合进行了扫描。迭代器的内容按顺序写入，首先是字符串“Quercus!”，然后是 Java Date 对象，最后是包含 5 个数字的 PHP 数组。

在浏览器中运行 http://127.0.0.1/1706.php，页面输出内容如下：

```
array(0) { }
string(8) "Quercus!"
resource(Sun Oct 04 00:00:00 CST 1970)
array(5) { [0]=> int(1) [1]=> int(2) [2]=> int(3) [3]=> int(4) [4]=> int(5) }
```

7. 在 D:\howwe\hocat\webapps\ROOT 下新建 1707.php

代码如下：

```
<?php
$system = new Java("java.lang.System");
var_dump($system);
echo("</br>Current time: ".
$system->currentTimeMillis()."</br>");
?>
```

代码分析：静态方法和字段直接使用类名称进行直接访问，访问静态字段与此类似。在浏览器中运行 http://127.0.0.1/1707.php，页面输出内容如下：

```
resource(null)
Current time: 1283696662968
```

8. 在 D:\howwe\hocat\webapps\ROOT 下新建 1708.php

代码如下：

```
<?php
try {
$system = new Java("java.lang.System");
    $system->getProperty(FALSE);
} catch (JavaException $exception) {
    echo "Cause: ".$exception->getCause();
}
?>
```

在浏览器中运行 http://127.0.0.1/1708.php，页面输出出错信息：

```
java.lang.IllegalArgumentException: key can't be empty
```

可见，Quercus 在 PHP 中不能捕获 Java 异常。

14.5　Quercus 原理及展望

Quercus 能把 PHP 和 Java 结合起来，Java 用于后台的数据库查询、存储。而 PHP 作为最前端的页面展示，将具有良好的用户体验。Quercus 原理如图 14-5 所示：

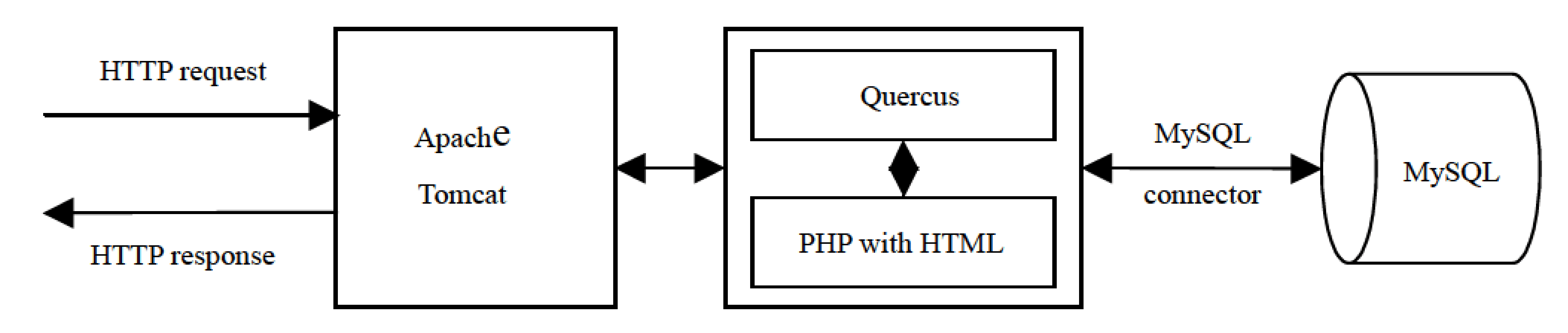


图 14-5　Quercus 原理

从图 14-5 可以看出，一个页面的请求为 HTTP request，用户通过该页面可以执行一些操作(增/删/改/查)，该请求被 Apache+Tomcat 接收。在 web.xml 中已定义用 Quercus Servlet 对象来解析 php 文件。该对象是一个 Java 的 Servlet，它提供与 Quercus 库的接口。在 PHP 文件中可以通过 MySQL connector(MySQL 连接器)来访问 MySQL 数据库。Quercus 一般通过 PDO(PHP Data Object，PHP 数据对象)以一种统一的方式，包括预处理语句等高级特性，提供数据库访问。

在实际应用中，还可以采用 PHP + Java + SOAP 或自定义 XML 传输协议的方式来实现 PHP 与 Java 的整合。

Quercus 在 Java + PHP 的整合方案中，更重要的是可以进行分布式计算，让 PHP + Java 整合方案在大型应用中可以更好地体验其伸缩性，如图 14-6 所示：

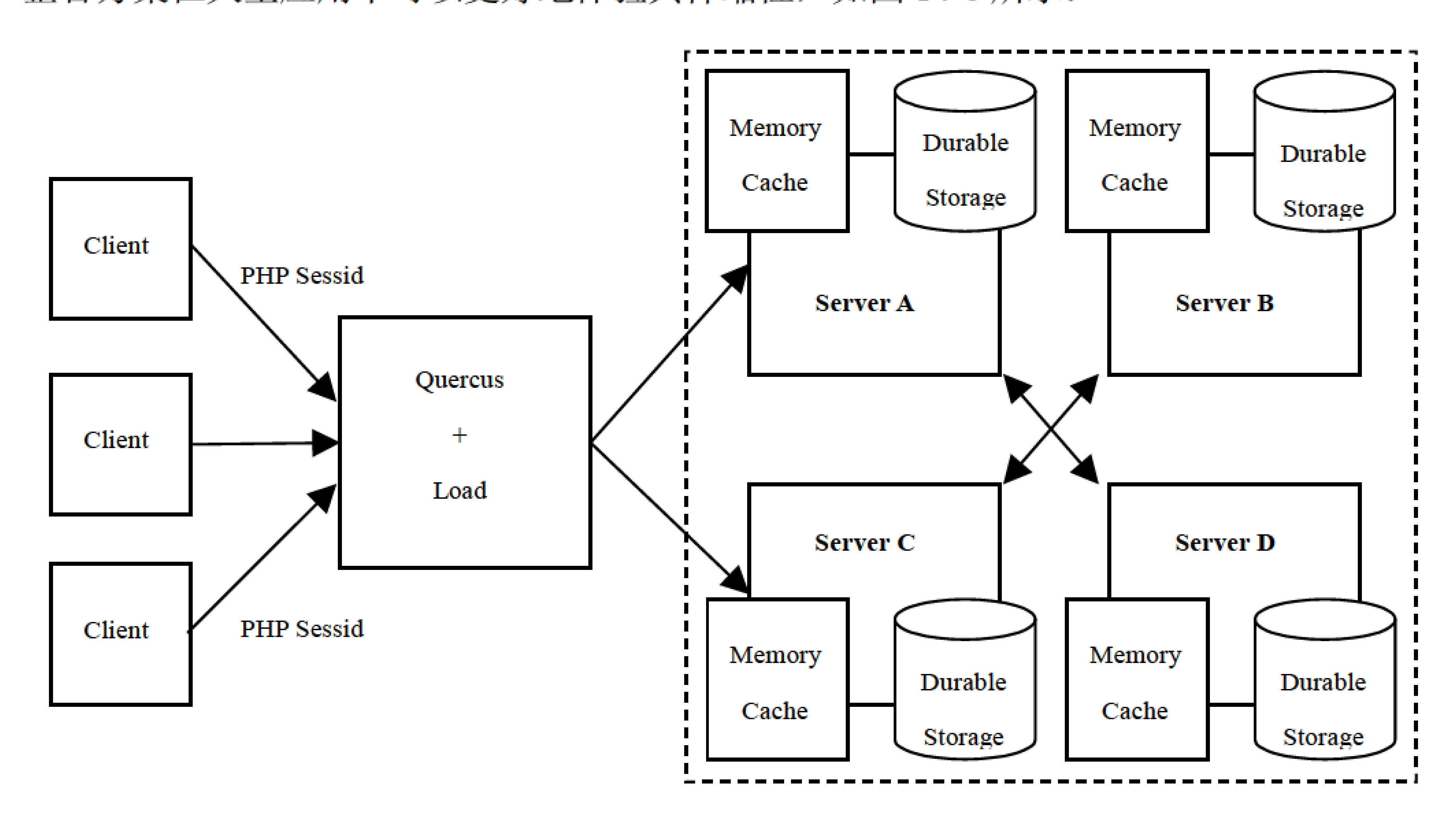


图 14-6　基于 J2EE 的大型 Quercus 应用

14.6 在 Quercus 环境下安装 PHP 应用

我们先去掉在“**14.3 配置 Quercus**”中已经配置好的环境，只使 php 目录支持 Quercus，步骤如下：

(1) 停止 D:\howwe\hocat 下的 hocat.exe，删除 D:\howwe\hocat\lib 下 inject-16.jar、javamail-141.jar、resin.jar 这三个 jar 文件。

(2) 打开 D:\howwe\hocat\conf 下的 web.xml，查找字符串“Quercus Servlet”，如存在，删除对应的“<servlet>”和“<servlet-mapping>”，内容如下所示：

```
<servlet>
  <servlet-name>Quercus Servlet</servlet-name>
  <servlet-class>com.caucho.quercus.servlet.QuercusServlet
  </servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>Quercus Servlet</servlet-name>
  <url-pattern>*.php</url-pattern>
</servlet-mapping>
```

(3) 在 D:\howwe\hocat\webapps 下建立目录“php”，将前面下载好的 quercus-4.0.8.war 用 winrar 解压到该目录下，下载地址为 http://caucho.com/download/quercus-4.0.8.war。

运行 D:\howwe\hocat 下的 hocat.exe，启动成功后，即完成 Quercus 环境的配置。

在浏览器中运行 http://127.0.0.1/php，页面输出内容如图 14-7 所示。

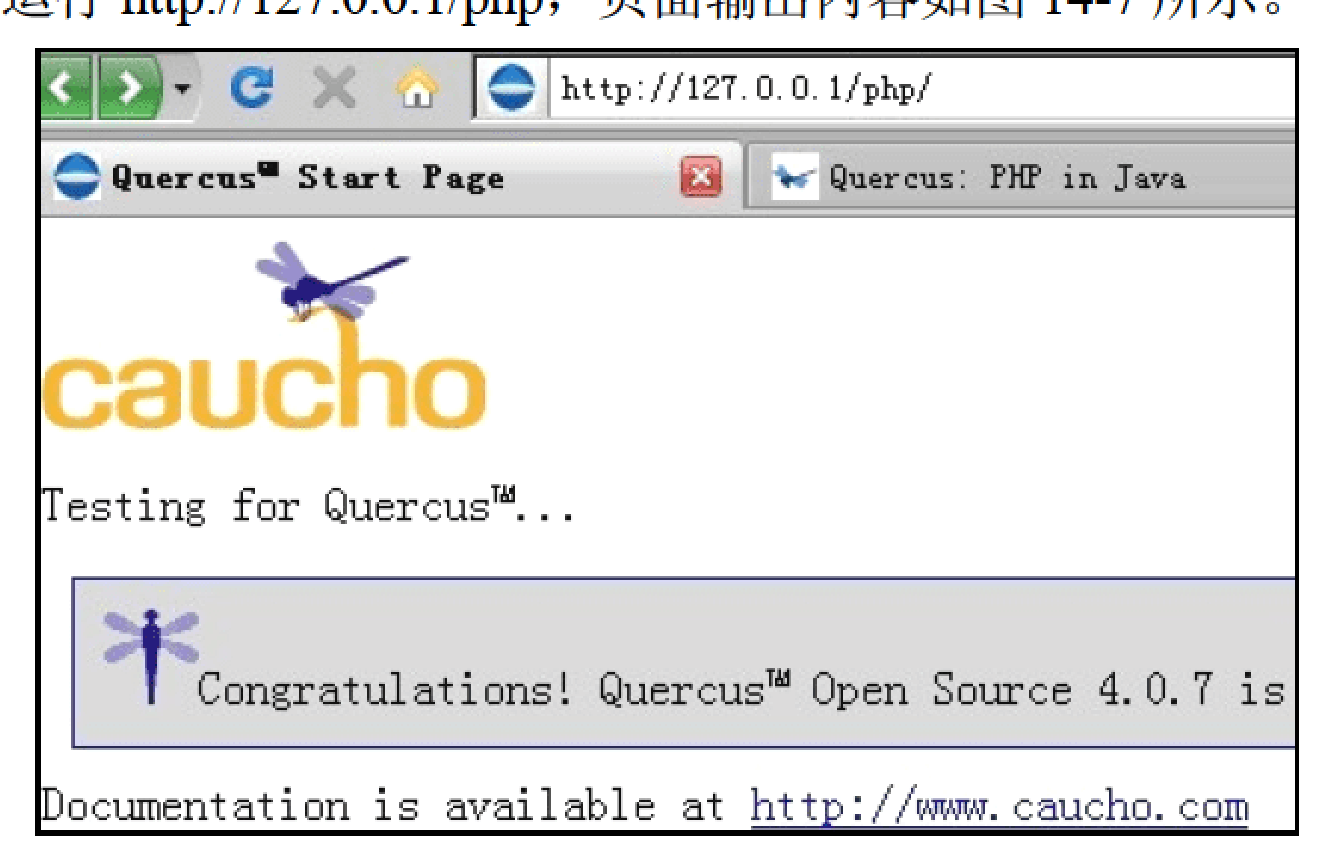


图 14-7 Quercus 单独配置在 php 目录下

接下来我们开始安装多个 PHP 应用。

1. 开源论坛软件 phpBB

(1) 在 D:\howwe\hocat\webapps\php 和 D:\howwe\mysql\data 下各建立目录 phpbb。

(2) 下载 phpBB3.0.7_PL1 中文版，下载地址为 http://www.phpbbchina.com/about，也可选择 D:\howwe\Other 下的 phpbb3.0.7_zh.zip。

(3) 将压缩包解压到 D:\howwe\hocat\webapps\php\phpbb 下，解压后，如有其他目录，请将无效目录去掉。

(4) 在浏览器中输入 http://127.0.0.1/php/phpbb，因为 phpbb 还未安装，故地址跳转到 http://127.0.0.1/php/phpbb/install/index.php 下，语言选择界面如图 14-8 所示：



图 14-8 phpBB 安装：语言选择

从图 14-8 可以看出 Quercus 对 phpBB 的支持存在小问题，选择第二项“??体中文”，单击“Change”按钮，则进入中文安装界面。选择“全新安装”->“继续进行下一步”->“开始安装”，在如图 14-9 所示的窗口“数据库设置”中填入对应内容：

数据库设置

数据库类型:	MySQL (使用 MySQLi 扩展)
数据库服务器地址，或 DSN: DSN 代表 数据源名称，它只与 ODBC 有关。	127.0.0.1
数据库服务器端口: 不用填写，除非您确定服务器监听一个非标准端口。	3306
数据库名称:	phpbb
数据库用户名:	root
数据库密码:	•
为数据库中的表格名称添加前缀:	bb_

继续进行下一步

图 14-9 phpBB 安装：数据库设置

所要填入的内容分别为“127.0.0.1”、“3306”、“phpbb”、“root”、“1”、“bb_”，填好后单击“继续进行下一步”。显示“数据库连接 连接检测：**连接成功**”后，再单击“继续进行下一步”。在如图 14-10 所示的窗口“管理员设置”中填入对应内容。

管理员设置

默认论坛语言:	简体中文
管理员用户名: 请输入一个3到20位的用户名。	admin
管理员密码: 请输入一个6到30位的密码。	••••••
确认管理员密码:	••••••
email联络地址:	a@b.cn
确认Email联络地址:	a@b.cn

继续进行下一步

图 14-10 phpBB 安装：管理员设置

所要填入的内容分别为“admin”、“admins”、“admins”、“a@b.cn”、“ a@b.cn”，填好后单击“继续进行下一步”。显示“管理员信息 检测管理员设置：**检测通过**”后，再单击“继续进行下一步”。在窗口“配置文件”中单击“继续进行下一步”。在窗口“高级设置”中单击“继续进行下一步”；如果数据库创建完成，则出现窗口“创建数据表”，单击“继续进行下一步”。出现“完成”窗口，提示“**移动或重命名 install 文件夹**”，处理完 install 文件夹，即可登录论坛。

在浏览器中输入 http://127.0.0.1/php/phpbb，phpBB 论坛首页如图 14-11 所示：



图 14-11 phpBB 论坛首页

依次单击“测试版面一” -> “欢迎来到 phpBB3”，再单击“编辑”，在编辑窗口内容的后面回车后输入“喻桃阳”三个字，再单击“提交”，部分页面截图如图 14-12 所示：

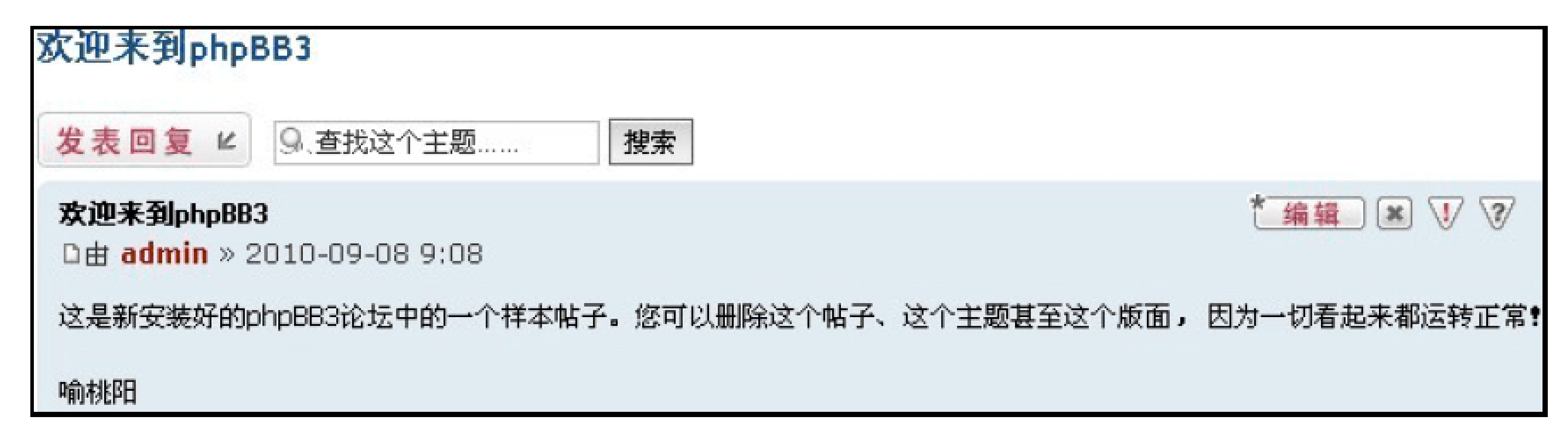


图 14-12　查看 phpBB 论坛主题

使用 D:\howwe\mysql\sqlfont 下的 SQL-Front.exe 打开 phpBB 中的表 bb_posts，字段 post_text 的内容如图 14-13 所示，其中字段 post_subject 为乱码。

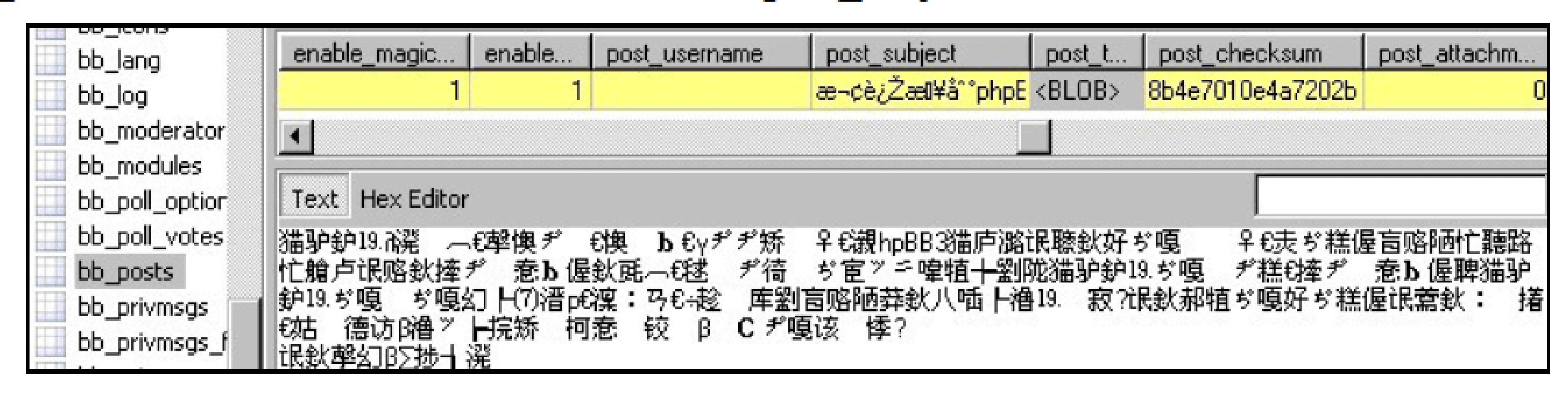


图 14-13　phpBB 论坛主题内容的对应字段

我们再单击图 14-12 中的“编辑”，在编辑窗口中内容的后面回车后输入“123456”，再“提交”，表 bb_posts 字段 post_text 的内容如图 14-14 所示，其中字段 post_subject 仍为乱码，但内容的最后多了“123456”6 个数字，可见该表对应论坛主题。

图 14-14　phpBB 论坛主题内容修改后的对应字段

我们再看看在 PHP 环境下安装的 phpBB，字段 post_subject 显示正常，如图 14-15 所示：

图 14-15　PHP 环境下的 phpBB 论坛主题

可见 Quercus 对 PHP 读取中文有问题，查官方网站的解释是 MySQL 驱动有问题，原文如下：“This is a known problem with the MySQL driver and Quercus. Unfortunately because of the unique string handling requirements of converting back and forth between Java and PHP and some issues in the MySQL driver, there are situations in which non-latin1 encodings will not work at the moment.”

2. 其他软件的安装

1) 安装 Ucenter 时，会出现一个 short_open_tag 未设置的错误，将安装文件中对 short_open_tag 的设置屏蔽掉，安装完后，访问主页时出现数据库连接错误。

在 WEB-INF 目录下增加一个 php.ini 文件，然后设置 short_open_tag = On，并修改 web.xml 文件启用 php.ini，这个错误就解决了。在<servlet-name>名为“Quercus Servlet”的 servlet 中，添加如下内容：

```
<init-param>
  <param-name>ini-file</param-name>
  <param-value>WEB-INF/php.ini</param-value>
</init-param>
```

将 php.ini 放在 D:\howwe\hocat\webapps\php\WEB-INF 下，即可安装 Ucenter，安装完成后，访问页面并输出如下内容：

```
Error:
Errno:0
SQL::
```

将在 PHP 环境中安装的 Ucenter 文件夹，即 D:\howwe\wwwroot\img 下的目录 api 复制到 D:\howwe\hocat\webapps\php 下。访问 http://127.0.0.1/php/api 时正常，但登录(密码为 admin)后从数据库读出的汉字为乱码，如图 14-16 所示，ID 为 8 的应用名仍为乱码。



图 14-16　Ucenter 读取中文为乱码

2) 安装 UCenter_Home 时不会报错，但安装完毕访问页面时，只要是从数据库读出的中文，就为乱码(方块)。

可见，PHP 在 Quercus 环境中直接访问 MySQL 时，无法支持中文。为了充分发挥两者整合的作用，必须采用 PHP 只处理用户界面(UI)，而 Java 负责数据处理的策略。下一节介绍这方面的范例。

浩为将启动一个学习小组来解决中文乱码的问题，相关内容请关注“浩为资源堂”的相应网址为 http://zyt.howwe.net/me.php?736，部分内容如图 14-17 所示。

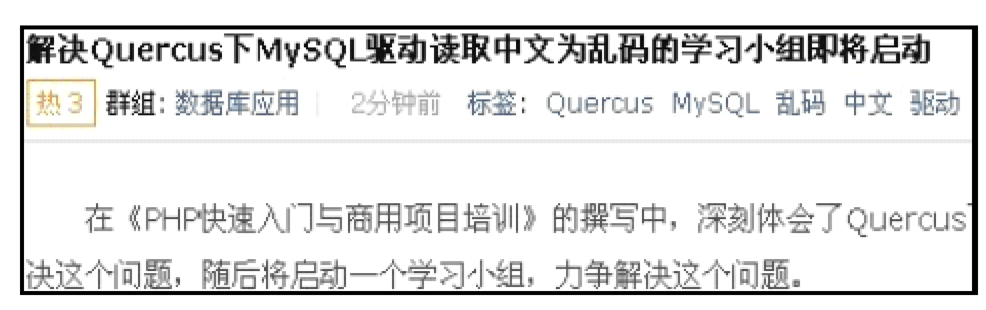


图 14-17　启动学习小组页面的部分内容

14.7　在 Quercus 环境下访问数据库

在讲解具体内容之前，先提供三个网址，前两个为 Quercus 的学习资料，第 3 个为 Quercus 官方主页。为了更好地掌握 Quercus，建议先自行阅读前两个网站上的内容。

1) quercus: php in java——http://www.caucho.com/resin-3.1/doc/quercus.xtp

2) uercus tutorials——http://caucho.com/resin/examples/quercus.xtp

3) Welcome to the home of Quercus——http://quercus.caucho.com

第 2 个网站中的“pdo”范例展示了如何在 PHP 中使用 php 代码 PDO 操作数据库，请自行仔细阅读，本节不再详细讲解。

注意：

数据库连接参数配置在 WEB-INF/resin-web.xml 的<database jndi-name = "jdbc/resin">处，而数据库连接仅用$pdo=new PDO("java:comp/env/jdbc/resin");即可。

下面简单介绍如何使用 PHP 整合 Java 来实现第 14 章和第 15 章介绍过的 HwCall 页面。

先将 D:\howwe\hocat\conf\Catalina\localhost\bak 下的 demo.xml 复制到上一层目录 localhost 下，稍等片刻，HoCAS 就能自动加载项目 demo，加载过程如图 14-18 所示。

```
D:\howwe\hocat\hocat.exe
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] -  hsqldb started
 ......
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] -  系统正在启动,
加载系统及SERVLET环境 ......
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] -  系统物理路径:
D:/howwe/hocat/webapps/../hocas-ok/webroot/
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] -  加载数据库连接
池 ......
registerDriver: driver[className=org.hsqldb.jdbcDriver,org.hsqldb.jdbcDriver@1cf
ad77]
DriverManager.getConnection("jdbc:hsqldb:hsql://127.0.0.1:19132/webapp")
    trying driver[className=sun.jdbc.odbc.JdbcOdbcDriver,sun.jdbc.odbc.JdbcOdbcD
river@ac97cc]
*Driver.connect (jdbc:hsqldb:hsql://127.0.0.1:19132/webapp)
    trying driver[className=com.mysql.jdbc.Driver,com.mysql.jdbc.Driver@114a306]

    trying driver[className=org.hsqldb.jdbcDriver,org.hsqldb.jdbcDriver@1cfad77]

getConnection returning driver[className=org.hsqldb.jdbcDriver,org.hsqldb.jdbcDr
iver@1cfad77]
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] - Detected server
 resin-tomcat
 INFO [ContainerBackgroundProcessor[StandardEngine[Catalina]]] - 系统启动完成 ..
....
```

图 14-18　HoCAS 自动加载 demo

在浏览器中输入 http://127.0.0.1/demo，登录窗口如图 14-19 所示：

图 14-19　HwCall 登录窗口

单击“登陆”按钮，待出现左边的菜单条后，单击“业务配置”下的“运营商管理”，页面截图如图 14-20 所示，URL 为 http://127.0.0.1/demo/smscOpDecl.jsp。



图 14-20　运营商管理窗口

下面将页面中的相关内容用 PHP 来实现。

1. 添加 Quercus 环境

为项目添加 Quercus 环境，步骤如下：

(1) 先停止 hocat.exe，在 D:\howwe\hocat\hocas-ok\webroot\WEB-INF 下的 web.xml 第一个<servlet>前添加如下内容，如存在则无需添加：

```
<servlet>
  <servlet-name>Quercus Servlet</servlet-name>
  <servlet-class>com.caucho.quercus.servlet.QuercusServlet
  </servlet-class>
</servlet>
```

```
<servlet-mapping>
  <servlet-name>Quercus Servlet</servlet-name>
  <url-pattern>*.php</url-pattern>
</servlet-mapping>
```

(2) 将 D:\howwe\hocat\webapps\php\WEB-INF\lib 下的三个 lib 文件：inject-16.jar、javamail-141.jar、resin.jar 复制到 D:\howwe\hocat\hocas-ok\webroot\WEB-INF\lib 下，重启 Hocat.exe 即可。

2. Eclipse 编辑项目

单击 D:\howwe 下的“快捷方式 到 eclipse.exe”启动 Eclipse，项目“hocas-ok”已经在项目列表中了，先加入 Quercus 的三个 jar：

(1) 先选中“hocas-ok”，单击右键，选择最下的“Properties”后，窗口如图 14-21 所示：

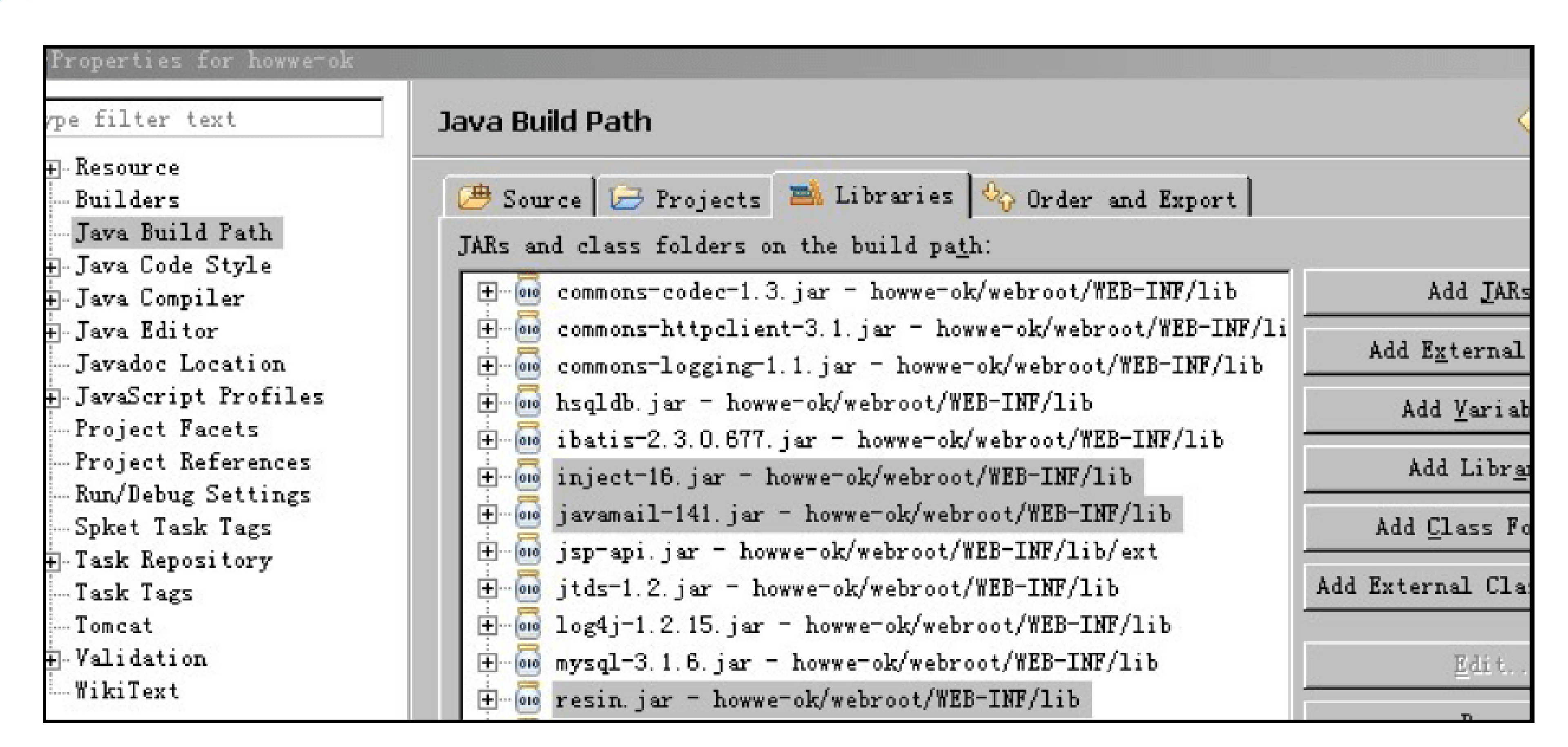


图 14-21　项目“hocas-ok”属性

单击按钮“Add JARS”，选择“hocas-ok”下的 webroot\WEB-INF\lib 下 Quercus 的三个 jar 后确定(单击 OK)，如已加入，则不会再列出。

(2) 选中“hocas-ok”下的“webroot”，单击右键，选择“New”->“PHP File”，输入文件名“smscOpDecl.php”，如图 14-22 所示，以默认方式生成 smscOpDecl.php。

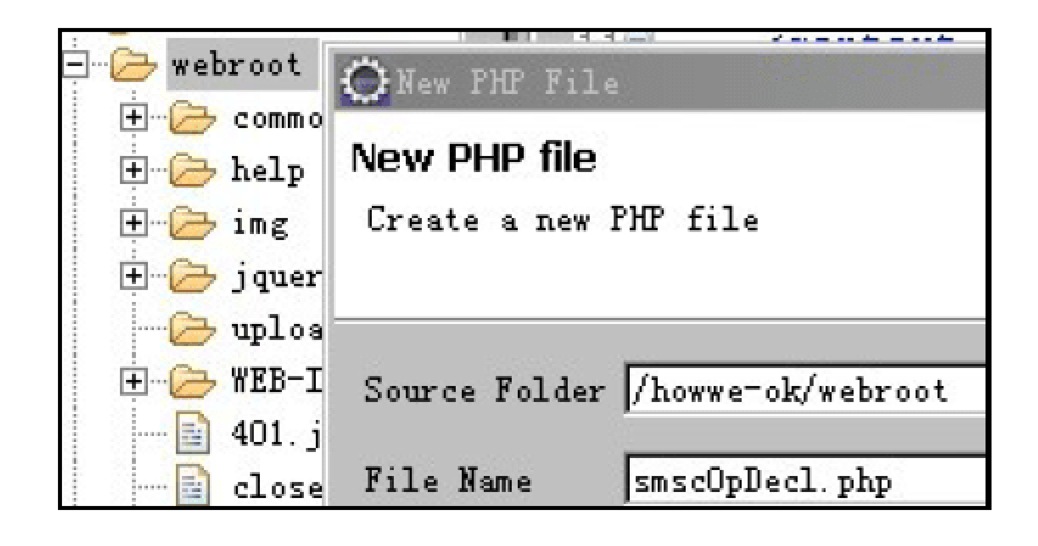


图 14-22　在项目“hocas-ok”的 webroot 下新建 PHP 文件

(3) 我们先看一个 PHP 调 JavaBean 的范例。先选中“hocas-ok”下的“src”，单击右键，选择“New”->“Other”->“Java” ->“Class”，新建一个 test.java，代码如下：

```
public class test {
    public int php_test(int i)
    {
        if (i > 0)
            return 31;

        if (i == 0)
            return 30;

        if (i < 0)
            return 29;

        return 100;
    }
}
```

输入后保存。选中“hocas-ok”下的“webroot”，单击右键，选择“New”->“PHP File”，新建一个“testJava.php”，代码如下：

```
<?php
$b=new Java("test");
$x=$b->php_test(5);

echo $x;
?>
```

保存后，在浏览器中运行 http://127.0.0.1/demo/testjava.php，在页面上输出 31，如图 14-23 所示：



图 14-23　PHP 调用 JavaBean

(4) 打开 smscOpDecl.jsp，将其另存为 smscOpDecl1.jsp，去掉 HTML 标签，代码如下：

```
<%@ page contentType='text/html; charset=GBK'%>
<%@ page import="net.howwe.core.dao.SmsDaocOpDecl"%>
<%@ page import="net.howwe.core.mo.SmsDatacOpDecl"%>
<%@ page import="java.util.ArrayList"%>
<%@ page import="java.util.Iterator"%>
```

```
<%
    SmsDaocOpDecl recDao = new SmsDaocOpDecl();
    ArrayList OdList = recDao.getSearch("",-1,"");
    net.howwe.util.deURL sUrl=new net.howwe.util.deURL();
    if (OdList.size() > 0) {
        Iterator itr = OdList.iterator();
        while (itr.hasNext()) {
            SmsDatacOpDecl rd = (SmsDatacOpDecl)itr.next();
            out.print(rd.getId()+" "+rd.getOpName()+" "+rd.getOpType()+" ");
            out.print(rd.getOpCode()+" "+rd.getWtime()+"
                      "+sUrl.EncS(rd.getOpName())+"<br>");
        }
    }
%>
```

参照此代码，在 smscOpDecl.php 中将其修改成相应的 php 代码，代码如下：

```
<?php
@header("content-Type: text/html; charset=gbk");
$b=new Java("net.howwe.core.dao.SmsDaocOpDecl");
$x=$b->getSearch("",-1,"");
$iterator = $x->iterator();
$d=new Java("net.howwe.core.mo.SmsDatacOpDecl");
$de=new Java("net.howwe.util.deURL");
while ($iterator->hasNext() == TRUE) {
     $d=$iterator->next();
     //var dump($d);
     echo $d->getId()." ".$d->getOpName()." ".$d->getOpType()." ";
     echo $d->getOpCode()." ".$d->getWtime() . " " .
               $de->EncS($d->getOpName()) . "<br>";
}
?>
```

在浏览器中运行 http://127.0.0.1/demo/smscOpDecl1.jsp，页面截图如图 14-24 所示：

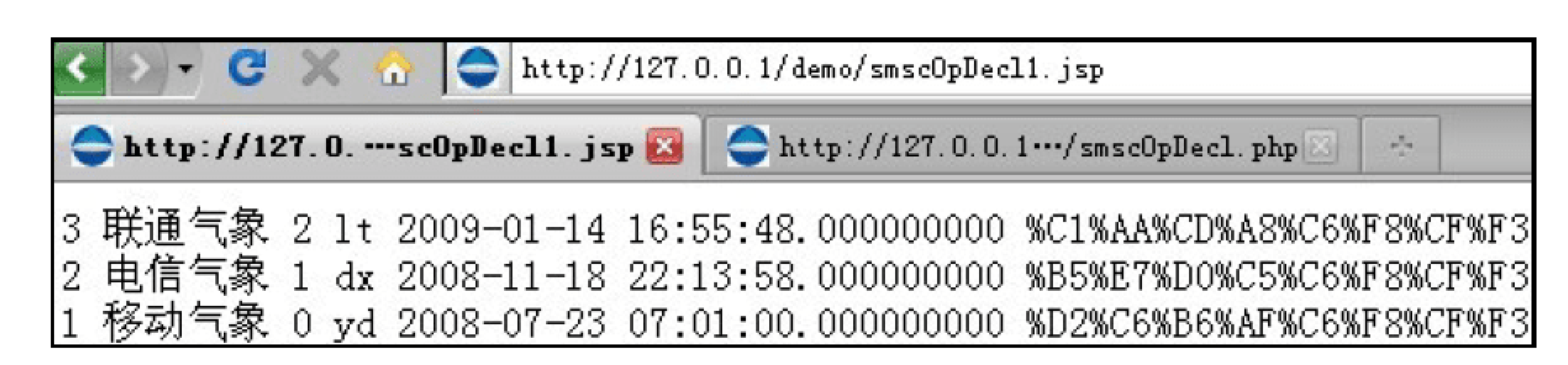


图 14-24　JSP 调用 JavaBean

在浏览器中运行 http://127.0.0.1/demo/smscOpDecl.php，页面截图如图 14-25 所示：

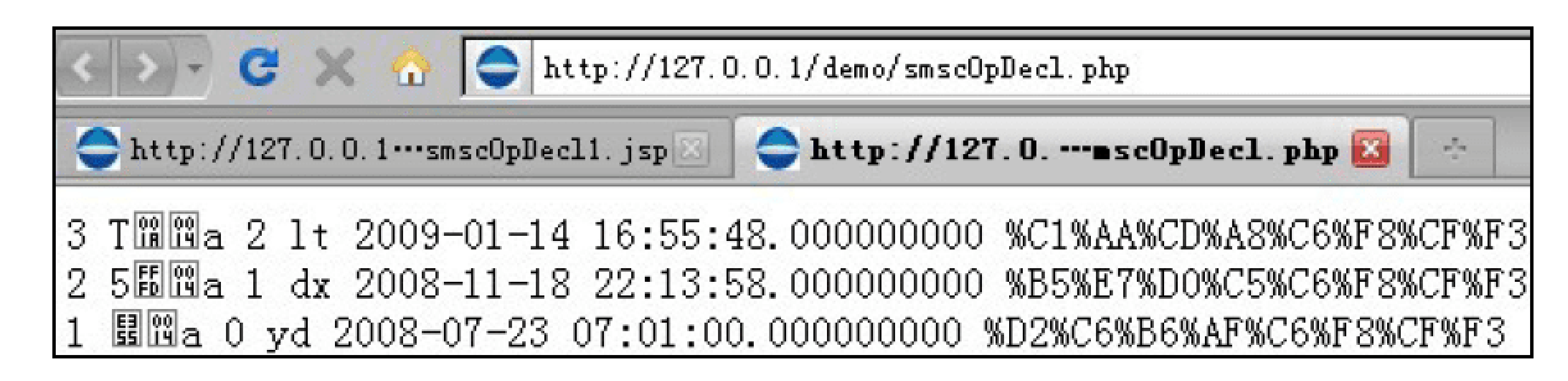


图 14-25 PHP 调用 JavaBean

从上面两图(图 14-24 和图 14-25)可以看出，JSP 和 PHP 调用同一 JavaBean，JSP 显示正常，而 PHP 显示不正常，后者的中文为乱码。

如何解决这一问题，这将是浩为学习小组要努力的一个方面，详情请见前一节的相关内容。

其实，UI(用户界面)只是一个显示数据的地方。在前面提到的 HoCAS，使用 jQuery 通过 Js 脚本来显示数据，这也许更是 PHP 与 Java 整合方案可思考的一个方向。具体不再展开，留给大家去思考吧！

14.8 在 sMash 环境下运行 PHP 应用

浩为整合包中的 howwe3.7z 已提供 WebSphere sMash(Zero)的运行环境，这是 IBM 提供的 PHP 整合 Java 方案，该方案给人的感觉是配置复杂，单单 Zero 的运行程序就 100MB，但项目的定位及远景很不错。整合包中 Zero 的版本为 1.1.1.4.31129。Zero 官方网站为 http://www.projectzero.org。

Zero 的全称是 Project Zero，是 IBM 启动的一个孵化项目，致力于为下一代动态 Web 应用程序提供敏捷的开发方法。Zero 引入了一种简单的环境，可以基于流行的 Web 技术创建、组装和执行应用程序，该环境包括一个面向 Groovy 和 PHP 的脚本运行引擎，并具有应用程序编程接口，这些接口可用于 REST 式服务、集成 Mashup 和生成富 Web 界面。

Zero 支持 PHP，在于其具有以下三个特点：

1) 敏捷性(Agility)：大多应用程序需要一直运行下去，即程序必须优化，而 PHP 的优化做得很好。

2) 内容(Content)：网上有不计其数的 PHP 资源，大多代码一般不太需要从头开始编写，搜索、剪切、粘贴即可完工，简称“剪贴式编程”(其实这又涉及敏捷性)。

3) 训练(Educated)：PHP 很容易自学，非计算机人员都很容易编写 PHP 代码，大多时候只是修改及利用别人的代码。几乎不用考虑复杂的代码，如内存管理、线程等。

尽管 Zero 的远景描述得很好，如部署 WebSphere sMash 后，PHP 程序员只需要一个火狐浏览器(Firefox)，就可以获得一个基本功能全面的 PHP 运行和开发环境，但目前可供

下载的版本中很难测试其功能。从 http://projectzero.org/download 下载的 WebSphere sMash DE v1.1.1，仅为安装下载器，运行 zero.bat 后，等一段时间后，下载内容仅为 Zero 的运行环境，1.1.1.4.31129 的下载内容如 howwe3.7z 所示。

注意：

howwe3.7z 已去掉 zero-repository 目录下的两个 expanded，因为运行 Zero 时，如果 expanded 不存在，可由相应的 modules 自动生成。

下面看看如何在 Eclipse 中使用 Zero。

由于在 howwe1.7z 中已包含 WebSphere sMash 插件，而 Zero 运行环境在 howwe3.7z 中，故在 D:\howwe\eclipse 下有个文件“使用说明.txt”，和 Zero 的相关内容如下。

WebSphere sMash(zero)的配置(zero 在 howwe3.rar 里)：

(1) 选择 **Window->Preferences->WebSphere sMash**。

(2) 在 **Zero home** 处选择目录 d:\howwe\zero。

(3) 在 **Module group** 处选择 experimental。

注意：

howwe3.rar 为 WinRAR 生成的压缩包，现已改用 7-zip 来压缩(后缀为 7z，但可用 WinRAR 来解压)，因为后者压缩率更高。

WebSphere sMash 插件也可以通过如下方式来获取：

在 Eclipse 中选择 **Help->Install New software…**，在 **Work with** 中输入以下站点：https://www.projectzero.org/zero/indy.dev/latest/update/zero.eclipse.php/，再选择 **WebSphere sMash for PDT 2.0.x** 下的 **WebSphere sMash Feature** 和 **WebSphere sMash PDT 2 Feature**，再单击 **Next**，直至提示安装完毕。

运行 D:\howwe 下的 Eclipse，按上面的步骤配置好 Zero。

新建 PHP 项目，步骤如下：选择 **File->New->Project…->WebSphere sMash->WebSphere sMash PHP Application**，单击 **Next**，窗口如图 14-26 所示。输入项目名称(Project name)t1，位置(Location)为 D:\howwe\hocat\webapps\t1。项目创建完后的目录如图 14-27 所示。

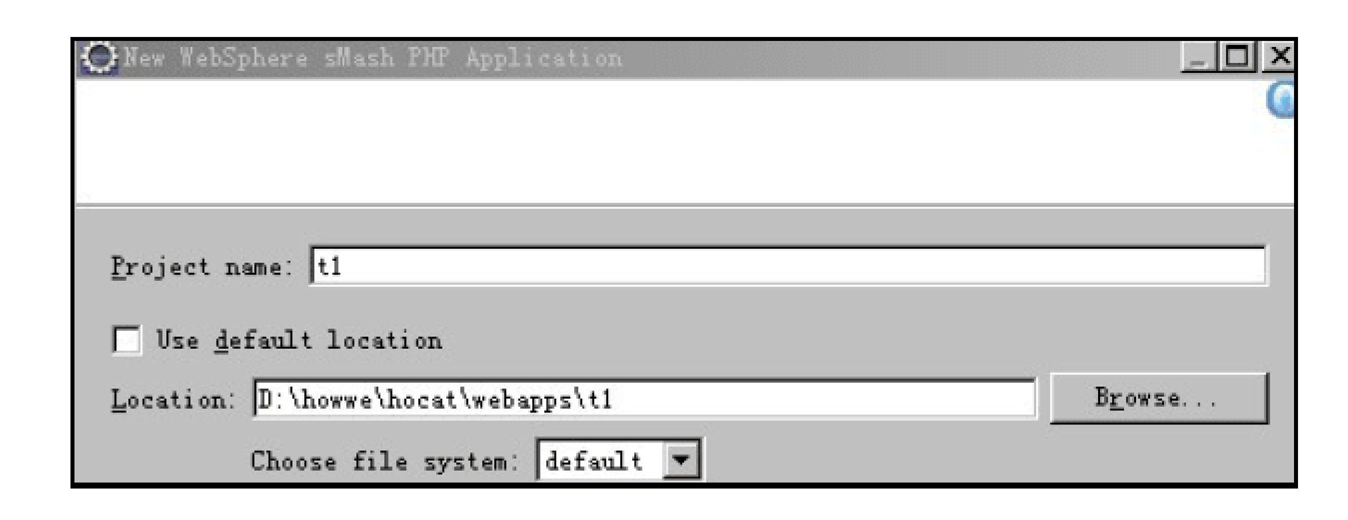


图 14-26　创建 WebSphere sMash PHP 应用

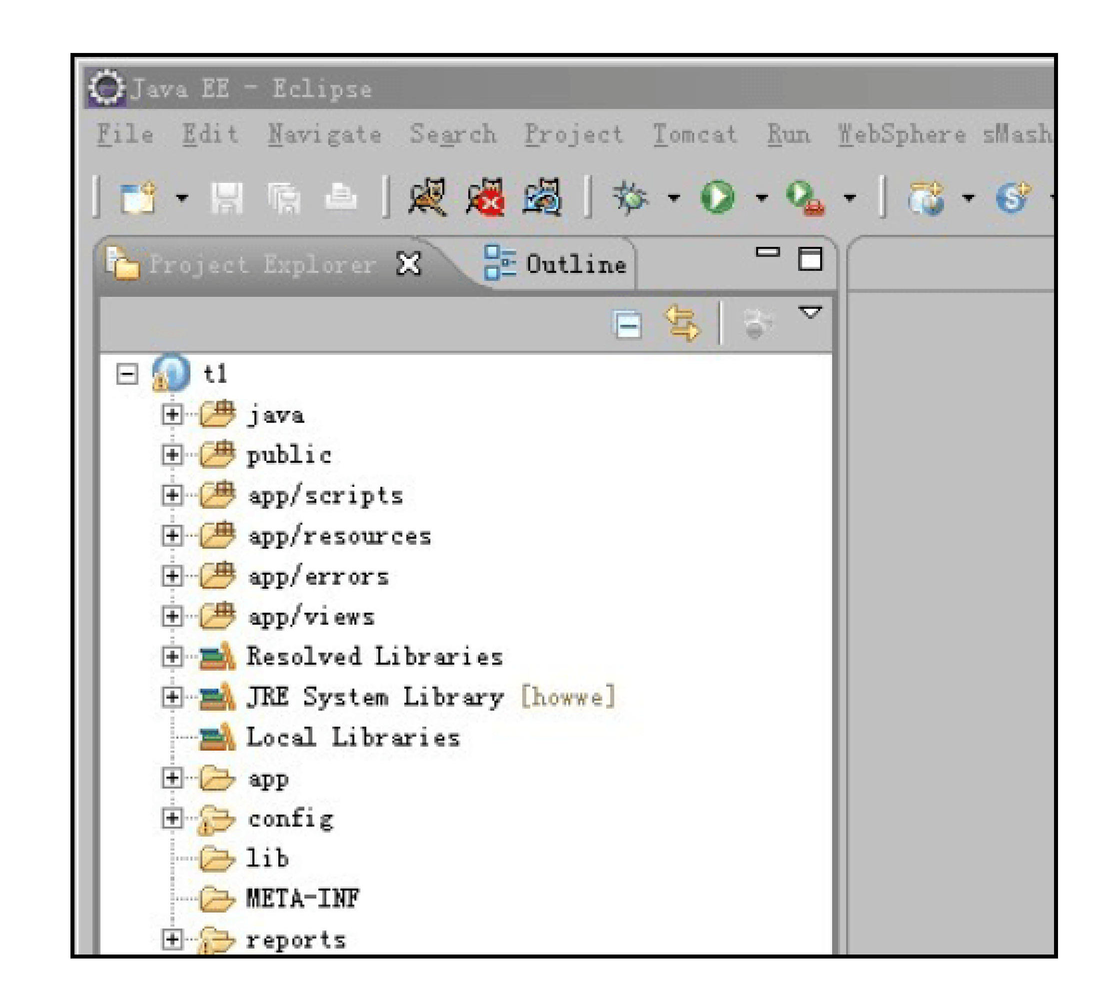


图 14-27　项目的目录

相关知识请自己 Google 或参考以下内容，本书不再展开：

1) Project Zero、WebSphere sMash 和 PHP 入门

http://www.ibm.com/developerworks/cn/web/wa-pz-intro/index.html

2) 在 WebSphere sMash 中集成 Java 和 PHP

http://www.ibm.com/developerworks/cn/websphere/library/techarticles/0809_phillips/0809_phillips.html

附　　录

常用的 JavaScript 语句

<1> document.write(""); 输出语句

<2> JS 中的注释为“//”

<3> 传统 HTML 文档的顺序是：document->html->(head,body)

<4> 一个浏览器窗口中的 DOM 顺序是：
window->(navigator,screen,history,location,document)

<5> 得到表单中元素的名称和值：
document.getElementById("表单中元素的 ID 号").name(或 value)

<6> 一个小写转大写的 JS：document.getElementById("output").value = document.getElementById("input").value.toUpperCase();

<7> JS 中的值类型：String，Number，Boolean，Null，Object，Function

<8> JS 中的字符型转换成数值型：parseInt()，parseFloat()

<9> JS 中的数值转换成字符型：(""+变量)

<10> JS 中的取字符串长度：(length)

<11> JS 中的字符与字符相连接：使用加号(+)

<12> JS 中的比较操作符：==(等于)、!=(不等于)、>、>=、<、<=

<13> JS 中声明常量用关键字 const，定义变量用关键字 var

<14> JS 中的判断语句结构：if(condition){}else{}

<15> JS 中的循环结构：for([initial expression];[condition];[upadte expression]) {inside loop}

<16> 循环中止的命令：break

<17> JS 中的函数定义：function functionName([parameter],...){statement[s]}

<18> 当文件中出现多个 form 表单时，可以用 document.forms[0],document.forms[1]来代替

<19> 窗口：打开窗口 window.open(), 关闭窗口 window.close(), 窗口本身 self

<20> 状态栏的设置：window.status="字符";

<21> 弹出提示信息：window.alert("字符");

<22> 弹出确认框：window.confirm();

<23> 弹出输入提示框：window.prompt();

<24> 指定当前显示链接的位置：window.location.href="URL"，

<25> 取出窗体中的所有表单的数量：document.forms.length，

<26> 关闭文档的输出流：document.close();

<27> 字符串追加连接符：=

<28> 创建一个文档元素：document.createElement(),document.createTextNode()

<29> 得到元素的方法：document.getElementById()；

<30> 设置表单中所有文本型成员的值为空：

```
var form = window.document.forms[0]
for (var i = 0; i<form.elements.length;i ){
if (form.elements.type == "text"){
form.elements.value = "";
        }
}
```

<31> 复选按钮在 JS 中判断是否选中：document.forms[0].checkThis.checked；(checked 属性代表是否选中，返回 TRUE 或 FALSE)

<32> 单选按钮组(单选按钮的名称必须相同)：取单选按钮组的长度 document.forms[0].groupName.length；

<33> 单选按钮组判断是否被选中也是用 checked

<34> 下拉列表框的值：document.forms[0].selectName.options[n].value；(n 有时用下拉列表框名称加上.selectedIndex 来确定被选中的值)

<35> 字符串的定义：var myString = new String("This is lightsword");

<36> 字符串转成大写：string.toUpperCase()；

字符串转成小写：string.toLowerCase()；

<37> 返回字符串 2 在字符串 1 中出现的位置：

String1.indexOf("String2")!=；-1 则说明没找到

<38> 取字符串中指定位置的一个字符：StringA.charAt(9);

<39> 取出字符串中指定起点和终点的子字符串：stringA.substring(2,6);

<40> 数学函数：Math.PI—返回圆周率，Math.SQRT2—返回开方，Math.max(value1, value2)返回两个数中的最大值，Math.pow(value1,10)返回 value1 的 10 次方，Math.round(value1)四舍五入函数，Math.floor (Math.random()*(n 1))返回随机数

<41> 定义日期型变量：var today = new Date();

<42> 日期函数列表：dateObj.getTime()得到时间，dateObj.getYear()得到年份，

dateObj.getFullYear()得到四位的年份，dateObj.getMonth()得到月份，
dateObj.getDate()得到日，dateObj.getDay()得到星期几，
dateObj.getHours()得到小时，dateObj.getMinutes()得到分，
dateObj.getSeconds()得到秒，dateObj.setTime(value)设置时间，
dateObj.setYear(val)设置年，dateObj.setMonth(val)设置月，
dateObj.setDate(val)设置日，dateObj.setDay(val)设置星期几，
dateObj.setHours 设置小时，dateObj.setMinutes(val)设置分，
dateObj.setSeconds (val)设置秒（注意：此日期时间从 0 开始计）

<43> FRAME 的表示方式：
[window.]frames[n].ObjFuncVarName，frames["frameName"].ObjFuncVarName，frameName.ObjFuncVarName

<44> arent 代表父亲对象，top 代表最顶端对象

<45> 打开子窗口的父窗口：opener

<46> 表示当前所属的位置：this

<47> 当在超链接中调用 JS 函数时用(javascript:)来开头，后面加函数名

<48> 在老的浏览器中不执行此 JS：<!-- //-->

<49> 引用一个文件式的 JS：<script type="text/javascript" src="aaa.js"></script>

<50> 指定在不支持脚本的浏览器中显示的 HTML：<noscript></noscript>

<51> 当超链和 onclick 事件都有时，老版本的浏览器转向 a.html，否则转向 b.html。
例如：dfsadf

<52> JS 的内建对象有：Array，Boolean，Date，Error，EvalError，Function，Math，Number，Object，RangeError，ReferenceError，RegExp，String，SyntaxError，TypeError，URIError

<53> JS 中的换行：\n

<54> 窗口全屏大小：<script>function fullScreen(){ this.moveTo(0,0);this.outerWidth= screen.availWidth;this.outerHeight=screen.availHeight;}window.maximize= fullScreen;</script>

<55> JS 中的 all 代表其下层的全部元素

<56> JS 中的焦点顺序：document.getElementByid("表单元素").tabIndex = 1

<57> innerHTML 的值是表单元素的值，例如<p id="para">"how are you"</p>，innerHTML 的值就是：how are you

<58> innerTEXT 的值和上面的一样，只不过不会把<em>这种标记显示出来

<59> contentEditable 可设置元素是否可被修改，isContentEditable 返回是否可修改的状态

<60> isDisabled 判断是否为禁止状态，disabled 设置禁止状态

<61> length 取得长度，返回整型数值

<62> addBehavior()是一种 JS 调用的外部函数文件，其扩展名为.htc

<63> window.focus()使当前的窗口在所有窗口之前

<64> blur()指失去焦点，与 focus()相反

<65> select()指元素为选中状态

<66> 防止用户在文本框中输入文本：onfocus="this.blur()"

<67> 取出该元素在页面中出现的数量：document.all.tags("div(或其他 HTML 标记符)").length

<68> JS 中分为两种窗体输出：模态和非模态。window.showModaldialog()，window.showModeless()

<69> 状态栏文字的设置：window.status='文字'
默认的状态栏文字设置：window.defaultStatus = '文字'

<70> 添加到收藏夹：external.AddFavorite("http://www.dannyg.com","jaskdlf");

<71> JS 中遇到脚本错误时不做任何操作：window.onerror = doNothing;
指定错误句柄的语法为：window.onerror = handleError;

<72> JS 中指定当前打开窗口的父窗口：window.opener，支持 opener.opener...的多重操作

<73> JS 中的 self 指的是当前的窗口

<74> JS 中状态栏显示内容：window.status="内容"

<75> JS 中的 top 指的是框架集中最顶层的框架

<76> JS 中关闭当前的窗口：window.close();

<77> JS 中提示是否确认的框：
if(confirm("Are you sure?")){alert("ok");}else{alert("Not Ok");}

<78> JS 中的窗口复位：window.navigate("http://www.howwe.net");

<79> JS 中的打印：window.print();

<80> JS 中的提示输入框：window.prompt("message","defaultReply");

<81> JS 中的窗口滚动条：window.scroll(x,y);

<82> JS 中的窗口滚动到位置：window.scrollby;

<83> JS 中的设置时间间隔：setInterval("expr",msecDelay)或 setTimeout 或 setInterval (funcRef,msecDelay)

<84> JS 中的模态显示在 IE 行，在 Netsscape Navigator 中不行：showModalDialog ("URL"[,arguments][,features]);

<85> JS 中退出之前使用的句柄：function verifyClose(){event.returnValue="we really like you and hope you will stay longer.";}} window.=verifyClose;

<86> 当窗体第一次调用时使用的文件句柄：onload()

<87> 当窗体关闭时调用的文件句柄：onunload()

<88> window.location 的属性：protocol(http:),hostname(www.example.com),

port(80)，host(www.example.com:80)，pathname("/a/a.html")，hash("#giantGizmo"，指跳转到相应的锚记)，href(全部的信息)

<89> window.location.reload()，刷新当前页面

<90> window.history.back()，返回上一页；window.history.forward()，返回下一页；window.history.go，返回第几页，也可以使用访问过的 URL

<91> document.write()，不换行输出；document.writeln()，换行输出

<92> document.body.noWrap=true;防止链接文字折行

<93> 变量名.charAt(第几位)，取该变量的第几位字符

<94> "abc".charCodeAt(第几个)，返回第几个字符的 ASCII 码值

<95> 字符串连接：string.concat(string2)，或用 “+ “进行连接

<96> 变量.indexOf("字符"，起始位置)，返回第一个出现的位置(从 0 开始计算)

<97> string.lastIndexOf(searchString[,startIndex])，最后一次出现的位置

<98> string.match(regExpression)，判断字符是否匹配

<99> string.replace(regExpression，replaceString)，替换现有字符串

<100> string.split(分隔符)返回一个数组存储值

<101> string.substr(start[，length])，取从第几位到指定长度的字符串

<102> string.toLowerCase()，使字符全部变为小写

<103> string.toUpperCase()，使全部字符变为大写

<104> parseInt(string[，radix(代表进制)])，强制转换成整型

<105> parseFloat(string[，radix])，强制转换成浮点型

<106> isNaN(变量)，测试是否为数值型